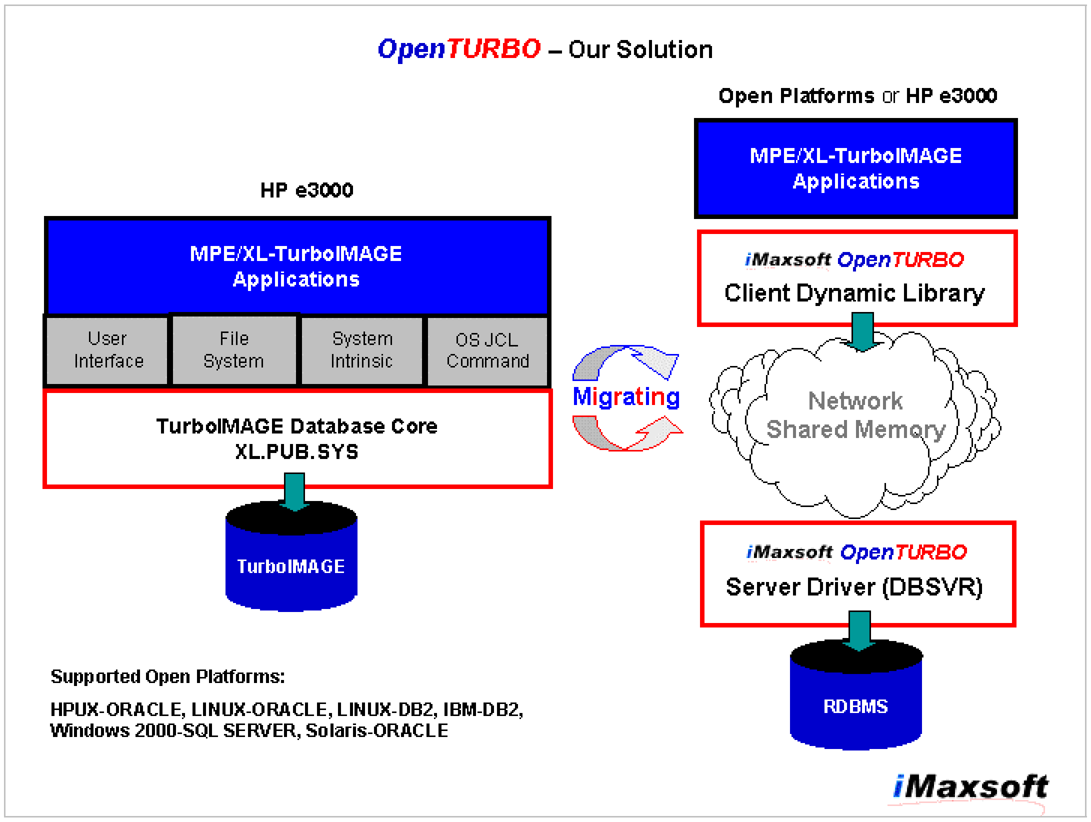
White Paper

IMAXSOFT OPENTURBO
HPe3000 Migration Methodology
and
Technology Overview

IMAXSOFT Corporation, California USA
April 13, 2005

OPENTURBO is designed to seamlessly migrate HP e3000 MPE/XL and TurboIMAGE based applications to an open platform, such as ORACLE on HP-UX, SQL SERVER 2000 on Windows 2000, ORACLE or DB2 on LINUX, and MySQL on UNIX. OPENTURBO reduces migration efforts from years to months, its risk free and speedy methodology can complete and certify 45%-75% of your migration in a few months. OPENTURBO preserves most of your critical business programs intact and keeps your business flow unchanged.

Since HP announced the discontinuation of HP e3000, users are trying to find ways to sustain their HP e3000 hardware and software investment as long as they can; at the same time, they are searching for either a total replacement or a migration path that can lead them into a 'longer life cycle' and 'more open' platform for their legacy applications. The end result is that a significant number of applications have to be migrated as it, due to 1) impossible to find an exact replacement, 2) costs too much to rewrite, 3) takes too long to rewrite, and most importantly 4) rewrite is too risky.

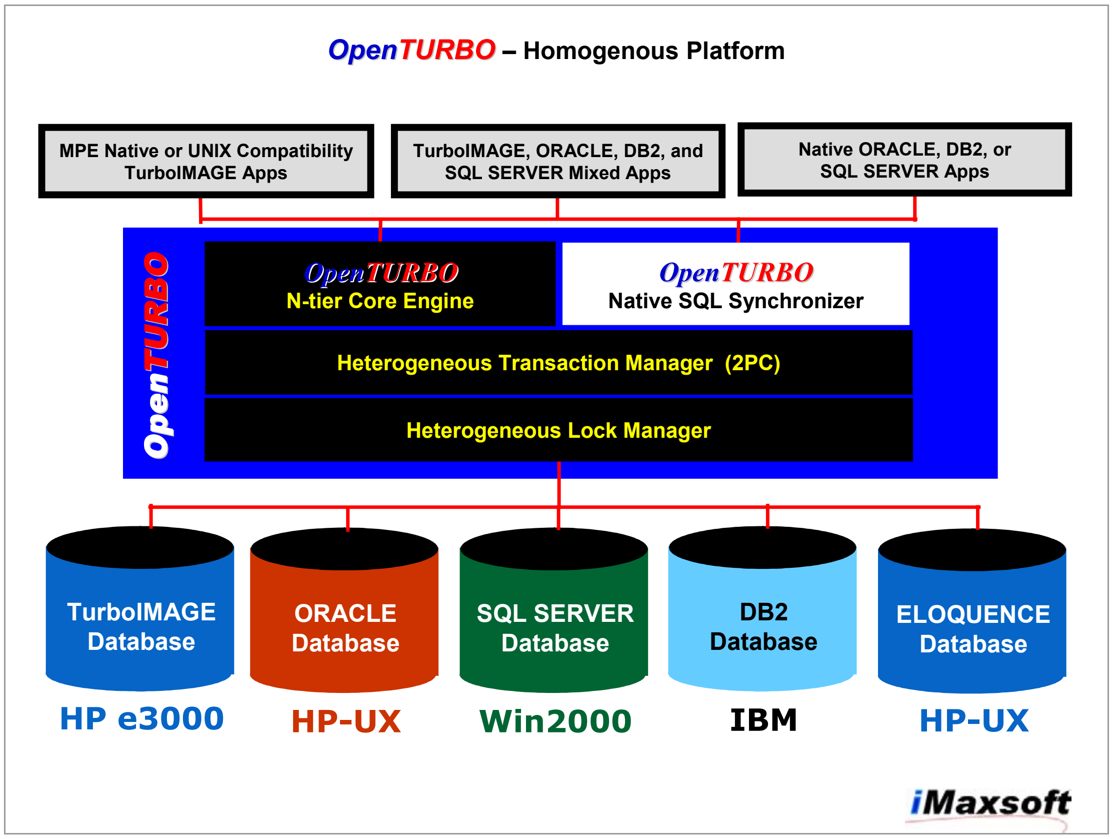

OpenTURBO – Our Solution
HP e3000
MPE/XL-TurboIMAGE Applications
User Interface
File System
System Intrinsic
OS JCL Command
TurboIMAGE Database Core XL.PUB.SYS
TurboIMAGE
Migrating
Open Platforms or HP e3000
MPE/XL-TurboIMAGE Applications
iMaxsoft OpenTURBO Client Dynamic Library
Network Shared Memory
iMaxsoft OpenTURBO Server Driver (DBSVR)
RDBMS
Supported Open Platforms:
HPUX-ORACLE, LINUX-ORACLE, LINUX-DB2, IBM-DB2, Windows 2000-SQL SERVER, Solaris-ORACLE
iMaxsoft

A full-range of HP e3000 migration consists of many components, depends upon your applications, the weight of each component may variant greatly; but in most cases, database and critical business logic are the two most troublesome items, unfortunately, these are essential to your core business operations and decision makings; therefore a migration methodology without automated assurance process to your data bank and business logic is meaningless. OPENTURBO is designed to specifically address these critical points, our methodologies and tools can guarantee a quick and success migration process, and a 100% assurance to migration results.

OPENTURBO Migration Methodology:

Our goal is to provide you a comprehensive migration platform which consists of migration automation toolset, data bi-directional synchronizer, MPE/XL and IMAGE emulators, applications cross-platforms and cross-databases translators, and varieties of foreign systems and applications interface adapters. OPENTURBO provides you an integrated and 100% parallelism migration environment for production, development and testing. You need no code-freeze, your databases, whether are productions or testing, IMAGE or ORACLE, are always synchronized in real-time, your applications, whether are original or migrated, can be deployed and re-deployed in phases with 100% transparent to your end-users, and you can roll-back your deployments to any fall-back recovery points with zero-down time.

Our migration methodology is RISK-FREE and PARALLELISM, our solution automates the migration process more than 80%; we modularized our tools and methodologies for varieties of migration needs, we are not only offer comprehensive toolsets to address 80% of your migration needs, we can also construct special tools for specific needs via our powerful building blocks, the foundation of IMAXSOFT's proven technology since 1987, and our string R&D experts, MPE/XL developers, IMAGE developers, IMAGE/SQL developers, ORACLE core developers, HP-UX developers, Windows/NT developers, Sybase developers, Solaria developers, COBOLII/BRW/C/C++ developers, and NS and TCP/IP developers; one of our customized solution for real-time replication is 'Bridge for HP 3000 IMAGE and IBM CICS DB2'.


OpenTURBO – Homogenous Platform
MPE Native or UNIX Compatibility TurboIMAGE Apps
TurboIMAGE, ORACLE, DB2, and SQL SERVER Mixed Apps
Native ORACLE, DB2, or SQL SERVER Apps
OpenTURBO
OpenTURBO N-tier Core Engine
OpenTURBO Native SQL Synchronizer
Heterogeneous Transaction Manager (2PC)
Heterogeneous Lock Manager
TurboIMAGE Database
ORACLE Database
SQL SERVER Database
DB2 Database
ELOQUENCE Database
HP e3000
HP-UX
Win2000
IBM
HP-UX
iMaxsoft

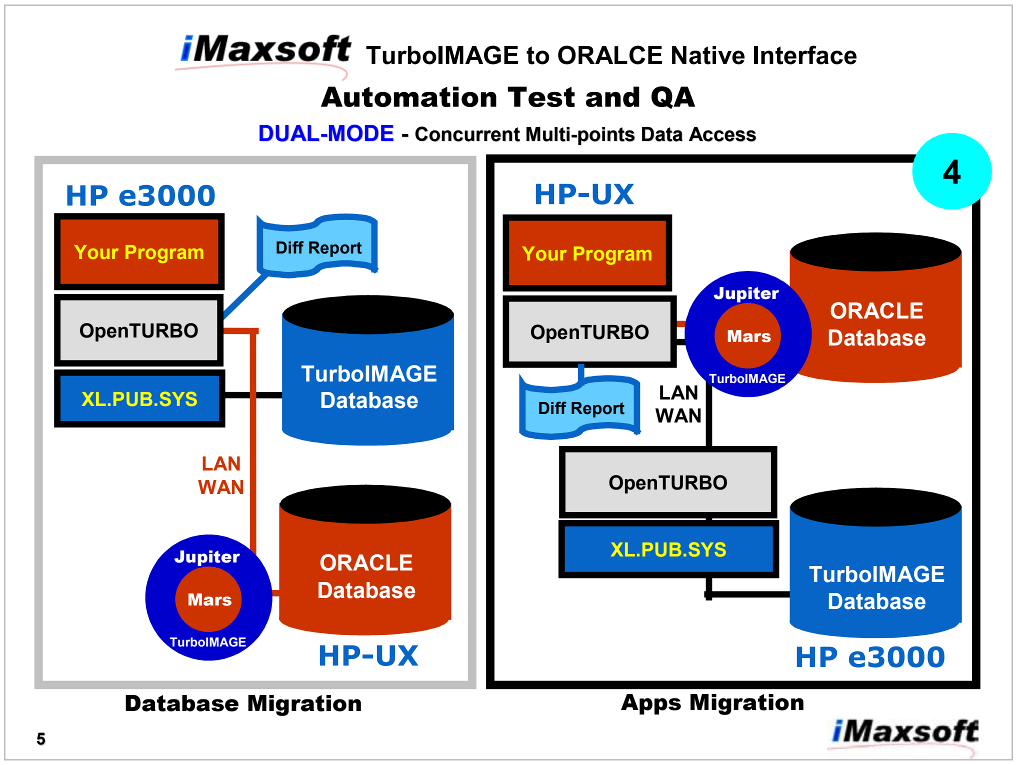
IMAXSOFT OPENTURBO provides a homogenous platform on heterogeneous environments. IMAXSOFT's migration platform is a mix of HP3000, HP-UX, LINUX, AIX, IMAGE, ORALCE, SQL SERVER, ELOQUENCE, and DB2; on our platform, you can direct your applications to run under any OS and to access one or more data sources concurrently at anytime; our goal is to enable you a true phase migration environment with 100% fall-back recovery capability.

IMAXSOFT’s parallelism technology allows co-existence of HP 3000, HP-UX and other OS applications, co-existence of emulated and native applications, provides gateway for a distributed database environment with two phase commit (2PC), and bi-directional synchronizer for real-time data replication and mirroring. The parallelism is not only a platform for migration, it is also a key path for your native migration in order to achieve 100% open.

Bi-directional Data Replication and Synchronization:

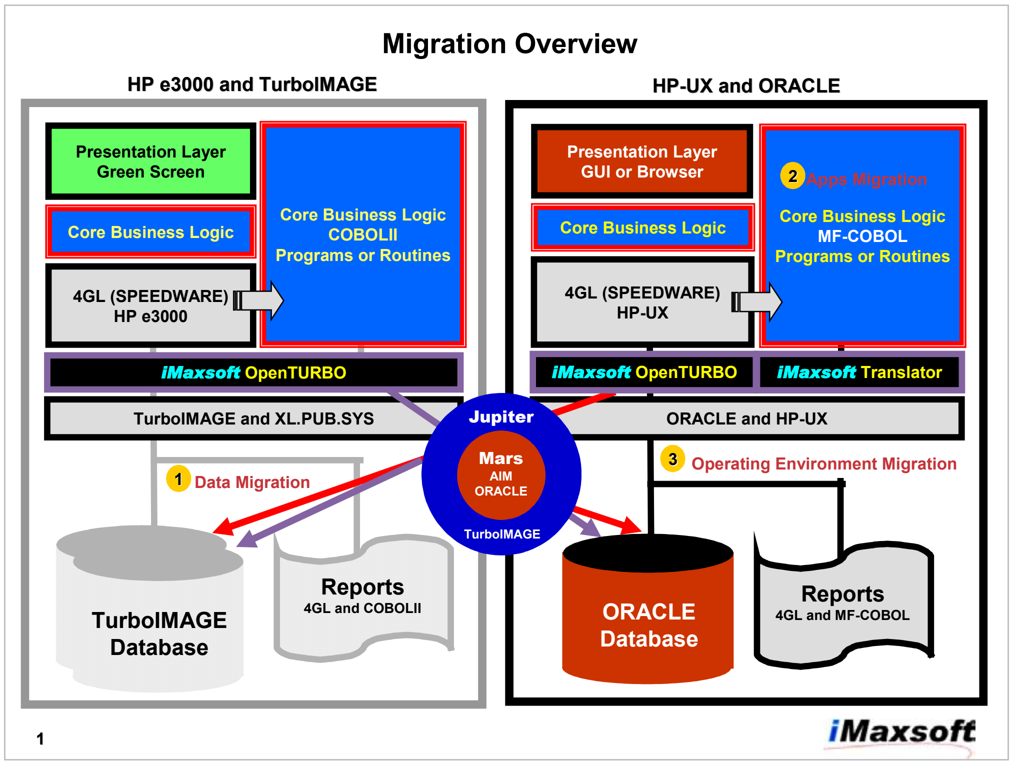
Transaction Level Real-time Data Replication (2PC):

OPENTRUBO Bi-directional Data Replicator is an application partition centric design; you can configure your application to run in single or multi database(s) mode, in multi databases mode you can configure transaction replication 2PC to target data sources only or mix of target data sources and log-files. 2PC to target data source only is so called real-time data synchronization or mirroring, whereas 2PC to mix of target data source and log-file is for fall-back recovery purpose.

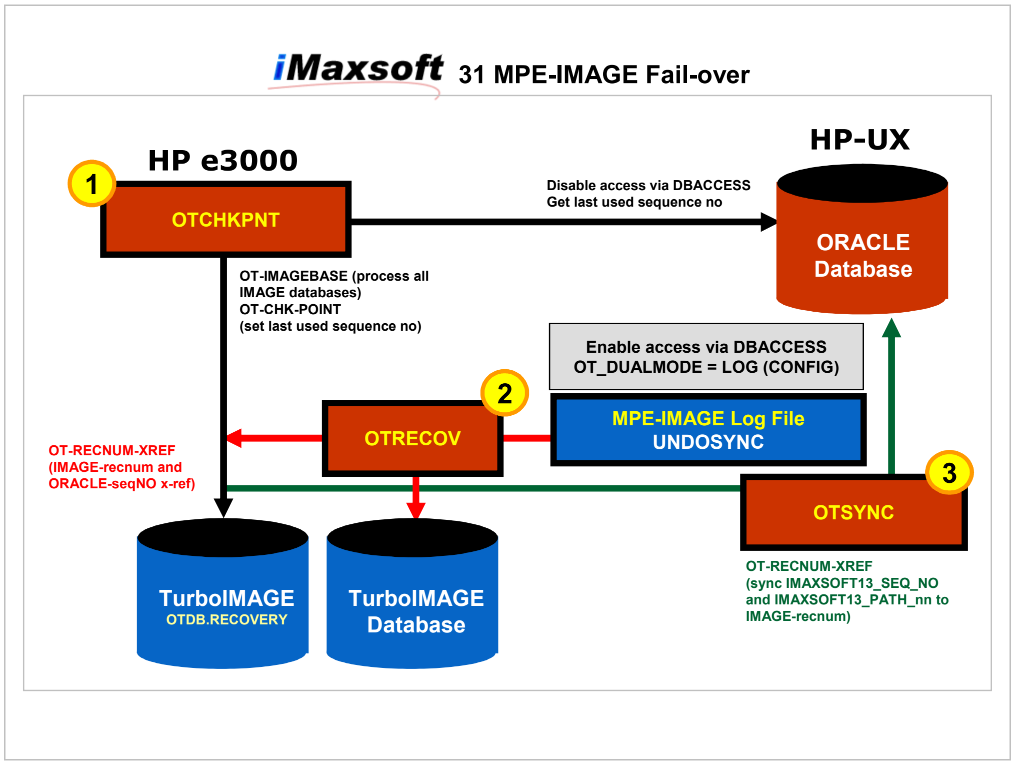


OPENTURBO Real-time Data Replicator translates TurboIMAGE calls into ORACLE SQL statements and applies updates to both ORACLE and TurboIMAGE simultaneously in 2PC (2 Phase Commit) mode or in off-line log-recovery mode.

DUAL-MODE is our internal code name for Bi-directional Real-time Data Replication, it is a layer in our core libraries and is completely transparent to MPE-TurboIMAGE development tools like Speedware, COGNOS, SUPRTOOL, TRANSACT, and COBOL. Therefore, you can run your MPE-TurboIMAGE applications from either HP e3000 or HP-UX and access TurboIMAGE and ORACLE concurrently for real-time data synchronization.

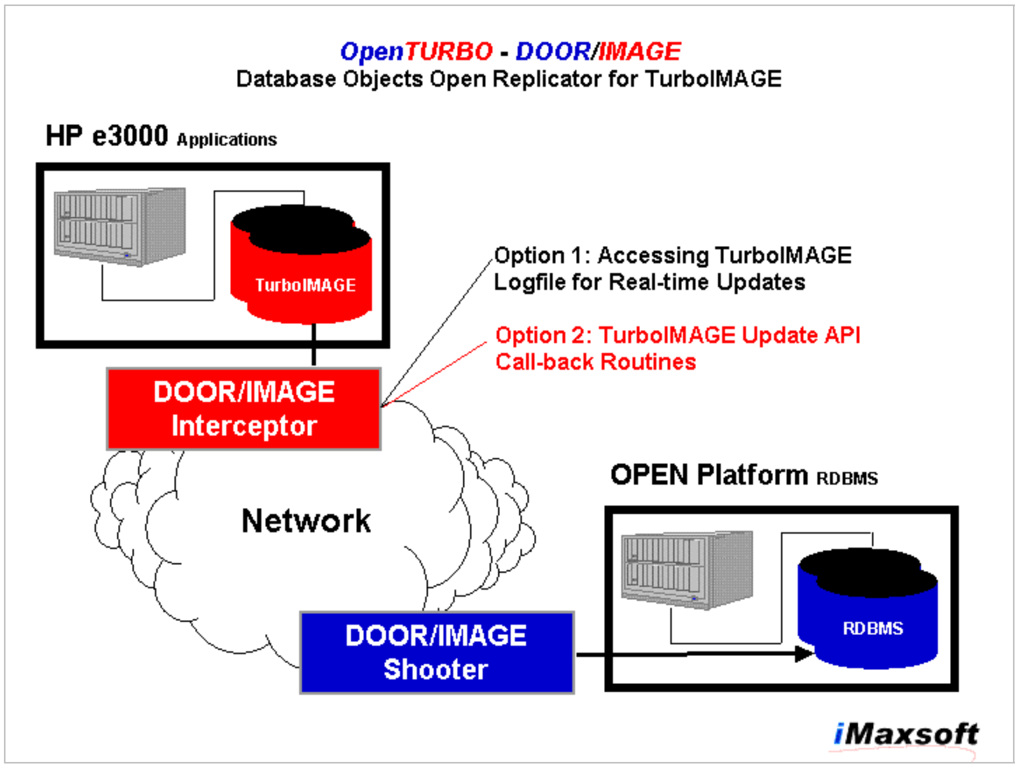


The above chart illustrates how data synching is done cross-platforms and cross-DBMS in real-time at application level. The application level data synching is required for bi-directional data synchronization in order to preventing infinite loop and racing problem.


iMaxsoft 31 MPE-IMAGE Fail-over
HP e3000
HP-UX
1
OTCHKPNT
Disable access via DBACCESS
Get last used sequence no
ORACLE Database
OT-IMAGEBASE (process all IMAGE databases)
OT-CHK-POINT (set last used sequence no)
Enable access via DBACCESS
OT_DUALMODE = LOG (CONFIG)
2
OTRECOV
MPE-IMAGE Log File UNDOSYNC
OT-RECNUM-XREF (IMAGE-recnum and ORACLE-seqNO x-ref)
3
OTSYNC
TurboIMAGE OTDB.RECOVERY
TurboIMAGE Database
OT-RECNUM-XREF (sync IMAXSOFT13_SEQ_NO and IMAXSOFT13_PATH_nn to IMAGE-recnum)

FAIL-OVER processor is mainly for off-line data replication and fall-back recovery, since it is off-line, the impact to the source servers are very minimum and major performance boost when transfer real-time sync to off-line sync.

DOOR – Database to Database Data Replication:


OpenTURBO - DOOR/IMAGE
Database Objects Open Replicator for TurboIMAGE
HP e3000 Applications
TurboIMAGE
Option 1: Accessing TurboIMAGE Logfile for Real-time Updates
Option 2: TurboIMAGE Update API Call-back Routines
DOOR/IMAGE Interceptor
Network
OPEN Platform RDBMS
RDBMS
DOOR/IMAGE Shooter
iMaxsoft

If you need to maintain both TurboIMAGE and ORACLE databases in-sync at all time, you want to maintain an exact mirror of TurboIMAGE in ORACLE on HP-UX, and you want 100% transparency to your applications, DOOR is solution; it is a database to database replication tool, it supports both real-time and office-line replication capabilities and it can replicate to multiple RDBMS on multiple server concurrently.

The difference between DUAL-MODE and DOOR, DUAL-MODE is implemented at transaction level, is application centric, and is TRUE real-time; in DUAL-MODE, all replications are done at transaction level autonomously, and you can configure DUAL-MODE at system, application or program levels; whereas, DOOR is independent to your applications, it replicates TurboIMAGE database updates to target RDBMS, you may provide mapping rules, but its main purpose is for database to database replication.

DOOR intercepts all TurboIMAGE updates, whether is from TurboIMAGE logging files or from TurboIMAGE API calls, transports and replicates changes to ORACLE on HP-UX in real-time or offline modes.

In DOOR, the OPENTURBO unique sequence number is mapped to TurboIMAGE internal record number, so, for serial scan DBGET mode 2 and 3, the data access orders are always in-sync.

DOOR is designed to support Zero-Down time database migration and is also used for replicating data from TurboIMAGE databases to multiple relational databases either on-line or off-line.

Zero-down time database migration methodology and DOOR:

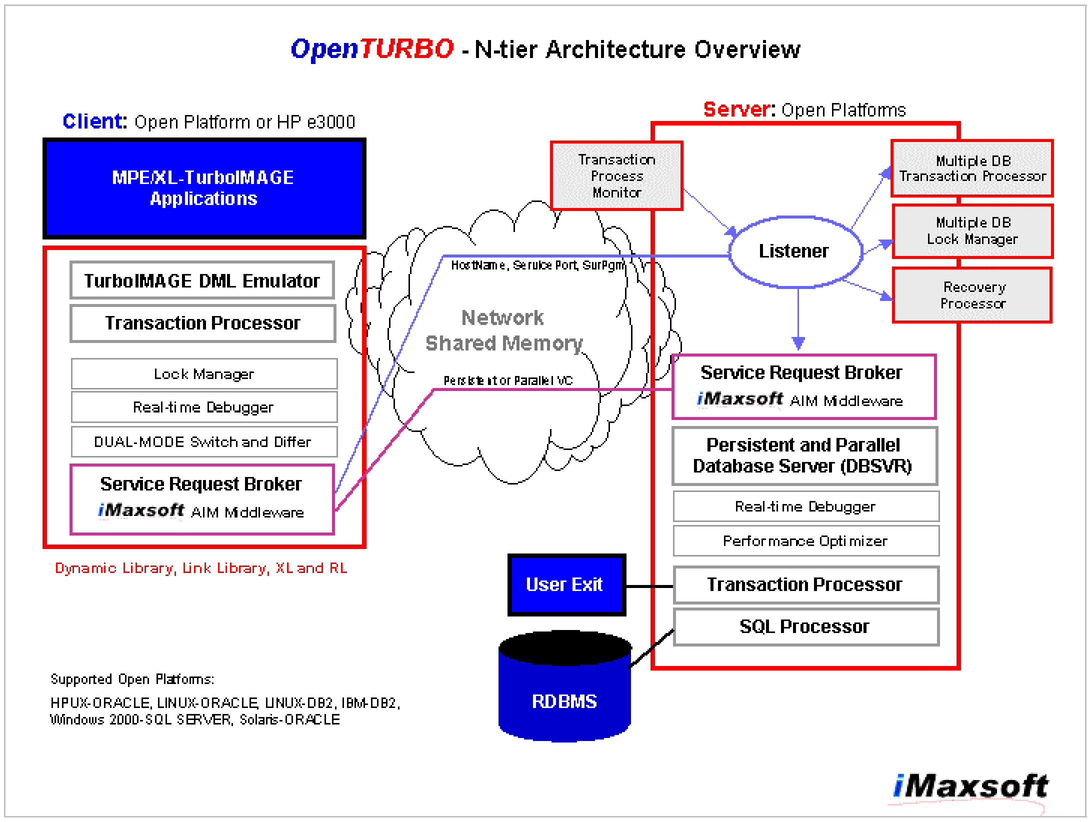
1) Do an on-line backup of your TurboIMAGE database and then immediately turn on TurboIMAGE Logging Facility or start DOOR Real-time Mode Interceptor.
2) Migrate the stored TurboIMAGE to ORACLE or others.
3) Run DOOR Shooter to replicate data to ORACLE from either TurboIMAGE Log-files or DOOR Intercepted Log-files until reach the most up-to-date Log-file.
4) Shut down TurboIMAGE, apply the most up-to-date Log-file to ORACLE – NOW YOU HAVE AN IDENTICAL MIRROR TurboIMAGE in ORACLE.

DOOR uses both TurboIMAGE Logging Facility and TurboIMAGE API Call-Interface Interception for data capture and can simultaneously replicate data to multiple RBDMS (ORACLE and SQL SERVER) on multiple servers. DOOR is a widely used for data warehousing.

DOOR Utilities:

DOORMAPPER	- DOOR Windows GUI Mapper
DOORMAP	- DOOR Replication Map File Generation
INTERCOT	- DOOR Interceptor, it intercepts TurboIMAGE updates either from TurboIMAGE logging files or directly from TurboIMAGE API calls, puts data into a queue for DOOR Shooter process.
SHOOTOT	- DOOR Shooter – it gets data from DOOR queue and transports them to one or many target relational databases on the Network.

OPENTURBO Architecture Overview:


OpenTURBO - N-tier Architecture Overview
Client: Open Platform or HP e3000
MPE/XL-TurboIMAGE Applications
TurboIMAGE DML Emulator
Transaction Processor
Lock Manager
Real-time Debugger
DUAL-MODE Switch and Differ
Service Request Broker iMaxsoft AIM Middleware
Dynamic Library, Link Library, XL and RL
HostName, Service Port, SurPgm
Network Shared Memory
Persistent or Parallel VC
Server: Open Platforms
Transaction Process Monitor
Listener
Multiple DB Transaction Processor
Multiple DB Lock Manager
Recovery Processor
Service Request Broker iMaxsoft AIM Middleware
Persistent and Parallel Database Server (DBSVR)
Real-time Debugger
Performance Optimizer
User Exit
Transaction Processor
SQL Processor
RDBMS
Supported Open Platforms:
HPUX-ORACLE, LINUX-ORACLE, LINUX-DB2, IBM-DB2, Windows 2000-SQL SERVER, Solaris-ORACLE
iMaxsoft

OPENTURBO is a n-tiers client-server architecture, it consists of TurboIMAGE emulator that is used by your existing HP3000 programs to access ORACLE on HP-UX as they were accessing TurboIMAGE on MPE/XL; a set of relational database transformation utilities that is used to replicate and move your entire TurboIMAGE database to ORACLE on HP-UX; and data integrity objects and toolsets that are used to support the co-existence of native relational database and OPENTURBO emulated applications.

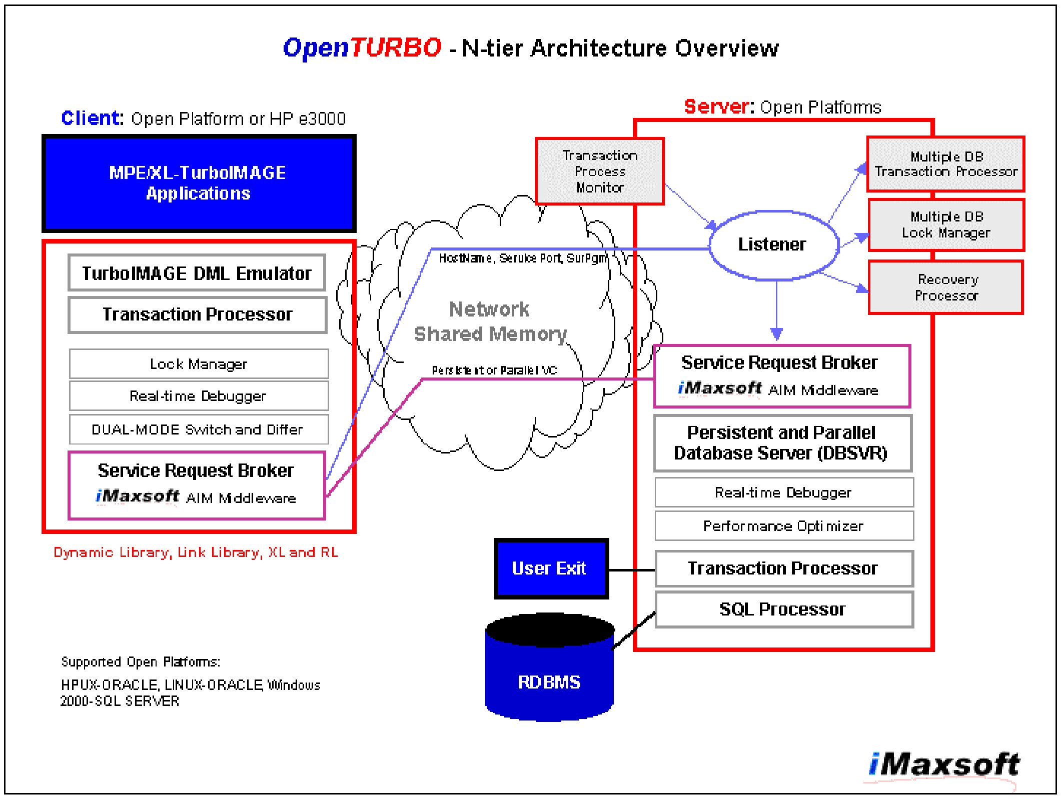
TurboIMAGE emulator contains a set of 100% emulated TurboIMAGE API calls and stored in a library which can be a XL on MPE/XL, a shared library on HP-UX, or a dll on Windows 2000. TurboIMAGE emulator libraries are also programming language dependent, for example BBASIC has its own library. You can migrated your MPE/XL TurboIMAGE programs without changing any database related calls and logic, simply link your migrated programs with our TurboIMAGE emulator library and run it. Our emulator will establish necessary connections and contexts for your program to access ORACLE on HP-UX. On the server, OPENTURBO has a set of processors to serve all requests from TurboIMAGE emulator client, perform ORACLE access, maintain context/thread and manage locks and transactions. OPENTURBO emulator server processors consist of Listener, Lanutil, SQL Server, Lock Manager, Recovery Processor, and Transaction Processor.

OPENTURBO relational database migration utilities transform your entire TurboIMAGE database structure objects, such as user passwords, access classes, keys, paths, sorts, path’s chronological order, unique identifier for each record, and data formatting and conversion, into a relational database ORACLE. OPENTURBO migration utilities validate all migrated TurboIMAGE objects and data afterwards for integrity and consistency.

OPENTURBO data integrity objects and toolsets guarantee the integrity of any future TurboIMAGE structure changes and the co-existence of native relational database applications and MPE/XL TurboIMAGE compatibility applications. OPENTURBO data integrity objects includes primary and foreign keys constrains, INSERT, DELETE and UPDATE triggers, row unique identifier and sequence, and lock manager objects. OPENTURBO toolsets allow you to change your ORACLE database and modify your programs in TurboIMAGE mode, this ensures that emulated applications and databases will always behave and act like TurboIMAGE until you replace them in native ORACLE mode.

OPENTURBO Components:

OPENTRUBO provides both client-server and direct modes for all libraries; the client-server mode allows you to partition your application to run on application server and access data from database server, it is designed to take full-advantage of N-tier client-server architecture, without changing you existing application you can take advantage of application server load balancing, distributed database environment and high availability; the direct mode allows you to run your application on a single hardware server and avoids any client-server related and network overheads in order to achieve maximum-performance and is mainly for batch jobs.


OpenTURBO - N-tier Architecture Overview
Client: Open Platform or HP e3000
MPE/XL-TurboIMAGE Applications
TurboIMAGE DML Emulator
Transaction Processor
Lock Manager
Real-time Debugger
DUAL-MODE Switch and Differ
Service Request Broker iMaxsoft AIM Middleware
Dynamic Library, Link Library, XL and RL
Supported Open Platforms:
HPUX-ORACLE, LINUX-ORACLE, Windows 2000-SQL SERVER
Transaction Process Monitor
HostName, Serivce Port, SurPgm
Network Shared Memory
Persistent or Parallel VC
Server: Open Platforms
Listener
Multiple DB Transaction Processor
Multiple DB Lock Manager
Recovery Processor
Service Request Broker iMaxsoft AIM Middleware
Persistent and Parallel Database Server (DBSVR)
Real-time Debugger
Performance Optimizer
User Exit
Transaction Processor
SQL Processor
RDBMS
iMaxsoft

TurboIMAGE/KSAM Data Manipulation Language (DML) Emulator:

OPENTURBO provides a set of libraries that 100% emulates the TurboIMAGE API Calls. If your HP e3000 MPE/XL and TurboIMAGE program does nothing but database accessing, data manipulation and business rules calculations, you can move the program straight across to the open platform, recompile it via the similar language compiler, and run it with OPENTURBO libraries.

Dynamic client and server libraries on HP-UX/ORACLE:

Library Name	Description
liblt	AIM Middleware and Debugging Facility Core Library
libot	TurboIMAGE/KSAM Core Library
libotdbg	TurboIMAGE/KSAM Core Library with Debugger
libsdk	SQL Server Core Library
libsdkc	SQL Client Core Library
libdbutx	DBSVR User Exit Routines Library
libti	TurboIMAGE Test Driver and Mapping Core

	Library

Dynamic client libraries on MPE/XL for HP-UX/ORACLE:

Library Name	Description
LTXL	AIM Middleware and Debugging Facility Core Library
OTXL	TurboIMAGE/KSAM Core Library
OTXLDBG	TurboIMAGE/KSAM Core Library with Debugger
OTQRY	TurboIMAGE/KSAM Core Library for QUERY.PUB.SYS
OTQRYDBG	TurboIMAGE/KSAM Core Library for QUERY.PUB.SYS with Debugger
DRIVER	Utilities Core Library
TIDRV	TurboIMAGE/KSAM Test Driver Core Library

Service Request Broker, AIM Middleware, and Database Server (DBSVR):

OPENTURBO Service Request Broker (SRB) consists of Service Request/Reply Library, Listener, and SQL Server Daemon.

OPENTURBO/SRB libraries are LTXL on MPE/XL and liblt on HP-UX or open platform. They are a set of routines used to move data and transactions across network to different machines and databases, to communicate among applications on the network, and to securely manage and control all traffics on the network.

The listener is a server daemon that accepts OPENTURBO connection requests from network or local loop-back, parses the request and dispatches OPENTURBO database server program (DBSVR).

The database sever program (DBSVR) is the OPENTURBO server program, it can only be spawned by OPENTURBO listener daemon. DBSVR is the core of OPENTURBO, it can be configured in 3 different modes: 1) persistent and direct connect, one DBOPEN per DBSVR, 2) multi-threaded persistent and direct connect, multiple DBOPENs for the same TurboIMAGE database per DBSVR, and 3) parallel and standby, multiple clients to one DBSVR, which is mainly for Web implementation but it also can apply to non-Web applications for better system throughput.

You can configure your MPE/XL and TurboIMAGE applications to connect to a TurboIMAGE clone on an open platform, or many TurboIMAGE clones on many different open platforms. Your TurboIMAGE clone can be distributed on a cluster machine or SAN, it can be mirrored and replicated to many other machines in real-time, it is a true relational database running on a native open platform, so it can take advantage of any new and advanced technologies, such as 128bit OS, Application Server, BLOB, JAVA, JDBC, and etc.

Transaction Processor:

OPENTURBO transaction processor supports both single TurboIMAGE database and multiple TurboIMAGE databases transactions (DBXBEGIN, DBXEND, and DBXUNDO).

By default, transaction is at TurboIMAGE call level, any update call (DBPUT, DBUPDATE, DBDELETE) forms a standalone transaction and it cannot be rolled-back. OPENTURBO default transaction handling is exactly like TurboIMAGE call level implementation and supports no transaction rollback.

For DBXBEGIN, DBXEND(COMMIT) and DBXUNDO(ROLLBACK) multiple database transaction, which is transaction across multiple TurboIMAGE databases in a single ORACLE instance, or transaction across multiple TurboIMAGE databases multiple ORACLE instances and multiple machines. Standard relational database COMMIT and ROLLBCK do **NOT** cross process context nor database instance boundaries, unless 2PC is used, but standard 2PC does not cover TurboIMAGE, therefore we create a multi-processes multi-instances and multi-servers COMMIT/ROLLBACK controller to protect data integrity and to support 2PC for atomic transaction across TurboIMAGE, ORACLE, DB2, SQL SERVER, ELOQUECE and others.

OPENTURBO transaction manager uses 2PC (2-Phase Commit) to process distributed transaction and replicated transaction, therefore database integrity and synchronization is guaranteed in all time regardless of how your database is partitioned.

Lock Manager:

TurboIMAGE is using abstract lock for concurrency control, and is inter-mixed with database open options. OPENTURBO lock manager provides identical functionality as TurboIMAGE DBOPEN and DBLOCK/DBUNLOCK do.

Our lock manager supports database, dataset and row level predicate locks. The client lock controller makes sure there is no lock conflict within the process, and the server lock controller controls and manages all lock requests at single and multi TurboIMAGE databases level.

Issues about co-existence of native relational and OPENTURBO compatibility applications:

TurboIMAGE uses abstract lock and the lock enforcement is at DBPUT, DBUPDATE, and DBDELETE call level, in order to synchronize relational database locks and OPENTURBO locks is quite challenge and is very expensive from performance perspective, so we create a LOCK Controller to control the co-existence conflicts. It is highly recommended that you set your locking strategies for your entire computing environment as early as possible and make it application centric and independent to your DBMS. Here are some suggestions:

1) **Avoid** update conflicts between compatibility OPENTURBO and native relational database applications; use phase migration method to un-plug and replace one applications at a time, and to avoid concurrent ORACLE and OPENTURBO updates to the same table or the same data object; TurboIMAGE and ORACLE use different locking mechanisms underline.
2) Apply the same TurboIMAGE **abstract** locking strategy to your relational database applications; an abstract lock object must be granted before any ORACLE or OPENTURBO updates.
3) All relational database applications invoke OPENTURBO lock object and use OPENTURBO lock manager via table update **trigger;** OPENTURBO lock manager enforces abstract and entity locking strategy and UPDATE INTEGRITY ENFORCEMENT which is a higher layer on top of TurboIMAGE and ORACLE lock managers.

UPDATE INTEGRITY ENFORCEMENT ensures a lock object is granted before conducting an update regardless it's origin, DB2, ORACLE, SQL SERVER, ELOQUENCE or TurboIMAGE; the lock object must be issued by the same update process and the update object must be a subset of the lock object. The update is a generic term of database update which includes INSERT, DELETE and UPDATE.

UPDATE INTEGRITY ENFORCEMENT trigger pseudo code (before table INSERT, UPDATE, and DELETE):

```
IF (ORACLE NATIVE) /* Non OPENTURBO Updates */
   IF (OT_LOCK_MANAGER(Update-Object) == EXIST and MINE)
         Proceed-Update
   ELSE
         Update-Object is locked, FAILURE
ELSE /* OPENTURBO Updates */
   Proceed-Update (OPENTURBO Updates)
```

SQL Processor:

OPENTURBO SQL Client Processor is in libraries OTXL and libot. It manages SQL access plan based on settings in CONFIG file; it synchronizes OPENTURBO and relational database transactions; it controls relational database CURSOR open, keep, switch, and close; it maps item list to column list; it builds the SQL query and update statements; and it converts TurboIMAGE data to/from relational data via OPENTURBO universal data-type formatter which is OS transparent and uses network byte order transmission.

OPENTURBO SQL Server Processor, DBSVR, is OPENTURBO relational database access core program. It manages all relational database activities and contexts which includes database connection, transaction, cursor and query, update and security.

1) Connection:

OPENTURBO supports all TurboIMAGE DBOPEN modes from 1 through 8. OPENTURBO DBOPEN modes control is server based; so even you are in a distributed database environment, the controller can be designated to one central point or distributed to multiple points.

OPENTURBO ORACLE connection manager allows to consolidate multiple DBOPENs to one ORACLE login and in one ORALCE name domain for better ORACLE throughput and performance.

OPENTURBO can be configured to have multi-DBOPEN context, concurrency and transaction control run on OPENTURBO client sharing the same Application Server as your applications, and maintain ONLY one thread to ORACLE Database Server for maximum-performance and high-scalability.

2) Transaction:

TurboIMAGE Transaction:

```
DBXBEGIN (single or multiple databases)
   DBLOCK(base1)
   DBLOCK(base2)
   DBPUT(base1)
   DBUPDATE(base2)
   DBDELETE(base2)
```

DBXEND or DBXUNDO
DBUNLOCK() – not allowed before DBXEND or DBXUNDO

OPENTURBO Transaction:

OPENTURBO always maintains one active ORACLE transaction from the time you open your TurboIMAGE database.

Single database transaction, DBXBEGIN is mapped to the current active ORACLE transaction that is owned by the persistent DBSVR process, every DBXEND/DBXUNDO is mapped to COMMIT/ROLLBACK which terminates the current transaction and initiates another active transaction immediately afterwards, and this process is autonomy and is data integrity guaranteed.

Multiple databases transaction, DBXBEGIN is mapped to multiple ORACLE transactions, each transaction is managed independently by its corresponding persistent DBSVR process, every DBXEBD/DBXUNDO is mapped to multiple COMMIT/ROLLBACK which terminates each active ORACLE transaction and initiates active transaction for each DBSVR in a round-robin fashion, if any failures occur during this critical OPENTURBO COMMIT/ROLLBACK process, OPENTURBO transaction controller and recovery mechanism will kick in to undo the damage and re-try the autonomy action if it is recoverable.

You may configure OPENTURBO transaction manager to run coincide with relational database 2PC (2-Phase Commit) option to ensure your database integrity in a distributed environment.

In a single instance environment, the chance of data inconsistency occurrence is almost **none**, and you should always use relational database logging and recovery features in addition to OPENTURBO transaction recovery processor.

In a multi-instances and multi-machines environment, the issue is more complex and the recovery is more involved.

** OPENTURBO for ORACLE doesn’t support **dirty** read since ORACLE supports no dirty read (read un-committed). OPENTURBO transaction and isolation level control lays on top of relational database, we support whatever relational database vendor supports in term of dirty read.

3) Data Access - Cursor and Query:

TurboIMAGE Data Access Methods:

Methods	TurboIMAGE Call
Re-fetch	DBGET Mode = 1
Serial Scan	DBGET Mode = 2 and 3
Direct Fetch	DBGET Mode = 4
Chain Fetch	DBFIND and DBGET Mode = 5 and 6
Calculated Fetch	DBGET Mode = 7
Primary Calculated Fetch	DBGET Mode = 8

TurboIMAGE DBFIND Search Methods:

DBFIND Methods	Description
Mode 1 and 21	KEY = VALUE KEY LIKE VALUE% (B-Tree)
Mode 10	Not Certified by Vendor
Mode 4 and 24	KEY = VALUE

(B-Tree Searches)	KEY < VALUE KEY <= VALUE KEY > VALUE KEY >= VALUE KEY IN [VALUE1, VALUE2] KEY LIKE VALUE% KEY LIKE %VALUE%
Mode88	Free form of WHERE clause

OPENTURBO Data Access Methods:

TurboIMAGE Call	OPENTURBO Emulation
DBGET Mode = 1	Where IMAXSOFT13_SEQ_NO = current IMAXSOFT13_SEQ_NO
DBGET Mode = 2	Order by IMAXSFOT13_SEQ_NO ASC
DBGET Mode = 3	Order by IMAXSFOT13_SEQ_NO DESC
DBGET Mode = 4	Where IMAXSOFT13_SEQ_NO = input IMAXSOFT13_SEQ_NO
DBFIND & DBGET Mode = 5	Where key = VALUE Order by IMAXSOFT13_PATH_nn ASC
DBFIND & DBGET Mode = 6	Where key = VALUE Order by IMAXSOFT13_PATH_nn DESC
DBGET Mode = 7	Where key = VALUE
DBGET Mode = 8	Same as Mode 7
Mode78	Free form of WHERE clause

OPENTURBO Search Methods:

DBFIND Methods	OPENTURBO Emulation
Mode 1 and 21	Where key = VALUE (NUMBER) Where key = 'VALUE' (CHAR) Where key like 'VALUE' (CHAR)
Mode 10	Not Certified by Vendor
Mode 4 and 24 (B-Tree Searches)	Where key = VALUE Where key < VALUE Where key <= VALUE Where key > VALUE Where key >= VALUE Where key >= VALUE1 and key <= VALUE2 Where key like 'VALUE%' Where key like '%VALUE%'

OPENTURBO maps Automatic and Manual master datasets to relational B-Tree indexes, so all modes of DBFIND are mapped to standard relational database index scan.

Mode 1: Re-fetch:

The mode 1 of DBGET is simply performing a re-fetch via the current record's unique key, IMAXSOFT13_SEQ_NO.

Mode 2 and 3: Serial Scan:

Serial scan, during data migration, OPENTURBO will try to preserve your detail dataset overall record's chronological order by assigning a unique number via its raw dataset access order. OPENTURBO assigns a unique number, known as IMAXSOFT13_SEQ_NO, to all records; IMAXSOFT13_SEQ_NO can be either a unique sequence number starts from 1 or the internal TurboIMAGE record number.

TurboIMAGE ***dirty link list*** uses first free first use method for any updates, for those of frequently updated (DBDELETE and DBPUT) detail datasets, whose overall record's chronological order is not preserved at all. But, OPENTURBO only cares the record access order of serial scan not its true chronological order. Internally, OPENTURBO uses the unique sequence number to emulate TurboIMAGE serial scan access order by sorting IMAXSOFT13_SEQ_NO in ascending order for DBGET mode 2, and descending order for DBGET mode 3.

Note 1: the access orders from TurboIMAGE and ORACLE are always identical right after the migration.

Note 2: if you do have TurboIMAGE and ORACLE co-existence requirement (bi-directional real-time data replication); detail dataset updates occurred after database migration, TurboIMAGE uses dirty link list first free first use algorithm, whereas OPENTURBO uses chronological sequence number; therefore, their serial scan access orders will be no longer in sync from that point on forward.

Solution: Use DUAL-MODE and IMAGE-RECNUM options, which triggers OPENTURBO to use IMAGE-RECNUM for its internal chronological sequence number.

Note 3: OPENTURBO maintains chronological order for all records and paths in all tables.

Mode 4: Direct Fetch

The mode 4 of DBGET is performing a direct record fetch via a given record number, which is the IMAXSOFT13_SEQ_NO in OPENTURBO environment.

Mode 5 and 6: Cursor Fetch or Chain Get

Chain fetch, OPENTURBO preserves and migrates detail dataset path’s chronological order by assigning a sequence number (IMAXSFOT13_PATH_nn) in order to emulate a TurboIMAGE path’s forward link list. Path with sort key, OPENTURBO generates WHERE search condition based upon DBFIND argument and sets ORDER BY to ‘sort key + all fields after the sort key ascend’ for mode 5 DBGET and ‘sort key + all fields after the sort key descend’ for mode 6 DBGET. Path without sort key, OPENTURBO sets ORDER BY to OPENTURBO’s path sequence number, such as ‘IMAXSOFT13_PATH_nn ascend’ for path-nn in mode 5 DBGET and ‘IMAXSOFT13_PATH_nn descend’ for path-nn in mode 6 DBGET.

Due to,

i. the chain get is the most frequently used access method,
ii. the inter-changeable forward and backward chain get is allowed in the same data access path,
iii. in most of relational database, a cursor is always forward direction and doesn’t return the length (row count) of a cursor,
iv. the cursor resources are expensive and limited in a relational database,
v. overall OPENTURBO performance consideration,

so, OPENTURBO Cursor Manager assures you that critical relational database recourses are managed efficiently, such as CURSOR and TRANSACTION. OPENTURBO Performance Optimizer streamlines data access plans and enhances SQL queries performance. TurboIMAGE DBFIND and DBGET are translated to

SQL CURSOR, SQL FTECH, and SQL SELECT, based on application data access methods and patterns, memory pool utilization rates, and ORACLE access statistics; therefore, OPENTURBO translation technology is the key to your application performance. Proper dataset level performance configurations via OPENTURBO CONFIG file are significant for performance improvement and throughput enhancements.

OPENTURBO Cursor Manager and Performance Optimizer configurable conditions:

Condition 1:	The maximum length of a dataset's paths is predictable (see Path Analysis Report by running otANALYZER on HP-UX), and DBFIND mode 1 is used without wildcard.
Method:	OPENTURBO loads and caches the entire cursor pointers into memory, and processes cursor forward and backward movements in memory and flushed cache at on-demand basis; this BULK FTECH DBFIND method is efficient and has minimum traffics to/from ORACLE back-end server. All subsequent DBGET mode 5 or 6 is simply a unique index scan fetch.
Condition 2:	Applications performs only one direction DBGET only, never mix DBGET Mode 5 and Mode 6 in the same data access path.
Method:	OPENTURBO can take advantage of Cursor Manager bulk-fetch and pre-fetch features to reduce I/O and speed-up data transport.
Condition 3:	Applications ignores status[5-6] chain count, status[7-8] backward pointer, and status[9-10] forward pointer all together.
Method:	OPENTURBO can take advantage of Cursor Manager bulk-fetch, pre-fetch and no-cursor-movement features to reduce I/O and speed-up data transport significantly.

Mode 7 and 8: Calculated Get (HASH)

Hash index is not used in OPENTURBO, both mode 7 and 8 of DBGET are index scan via 'key = search value' condition.

4) Updates:

DBPUT is mapped to ORACLE INSERT:

1. You may only insert Manual and Detail tables.
2. The Automatic table is automatically maintained by OPENTURBO.
3. You may not insert Detail row that has no corresponding key in Manual table, which is controlled by PRIMARY and FOREIGN constraint.
4. All unique sequence number is maintained by OPENTURBO in chronological order automatically, which is managed by trigger, sequence, and index constraint.
5. All Detail paths are also maintained by OPENTURBO automatically via trigger, sequence and index constraint.

DBUPDATE is mapped to ORACLE UPDATE:

1. You may update any columns in Detail tables.
2. You may only update non-key columns in Manual tables.
3. You may not update Automatic tables at all.
4. CIUPDATE or NO CIUPDATE are the same to OPENTURBO, we implement it simply for syntactic emulation purpose. OPENTURBO maintains path related adjustments with or without CIUDPATE, which is identical to TurboIMAGE CIUPDATE path handling.

DBDELETE is mapped to ORACLE DELETE:

1. You may not delete a Manual entry, unless its foreign constrains are all eliminated.
2. The Automatic entry is **not** deleted automatically, you need to run otCLEAN for clean-up periodically. The Automatic tables are redundant to OPENTURBO, all Automatic datasets are mapped to ORACLE indexes for detail tables by default, unless you have applications that access Automatic dataset directly, otherwise you need no Automatic datasets and save tremendously from performance point of view.

   In most cases, Automatic tables are not migrated, therefore OPENTURBO does not spend extra performance cost for deleting entry automatically from Automatic table when the entry has no foreign constrains attached to it.

All rules for INSERT, DELETE and UPDATE, such as triggers, primary and foreign constrains, unique constrain and path control, are also applied to native ORACLE applications.

Proper locks must be provided before updates, refer to HP TurboIMAGE/XL Database Management System Reference Manual.

5) Security:

Semicolon is mapped to the root user of the specific TurboIMAGE view in ORACLE database; on HP e3000, semicolon is used in conjunction with the creator of TurboIMAGE database.

All other classes are mapped to users who have specific access authorities to columns and tables.

Internally, OPENTURBO uses SYNONYM name-space feature for connecting users, database objects, and securities at logical TurboIMAGE database level, which is a group of tables of an ORACLE instance.

6) Universal Data-Type Formatter:

OPENTURBO is using network byte order for data transport, which is OS independent. OPENTURBO supports NLS at both database and emulator levels.

OPENTURBO converts all data into BINARY mode in a half-word (16-bit) array, just like TurboIMAGE, regardless it is a character X, integer I or float R.

OPENTURBO supports all TurboIMAGE data types, I, J, E, R, P, Z, X, U, and K. OPENTURBO converts HP real R into IEEE float E during migration. OPENTURBO handles zoned decimal in both standard and non-standard positive number, which is that the last positive sign byte can be 0 through 9 or {, A through I.

7) DBSVR Modes:

OPENTURBO DBSVR has 3 modes, Dynamic Query Mode, Dynamic Query Multi-threaded Mode, and SQL Pre-process Mode. Each mode has its specific purpose, the Dynamic Query mode is the standard mode, which is used in most of the applications at all time. The Dynamic Query Multi-threaded mode is used for Web applications, and the SQL Pre-process mode is only for super high volume transactions applications and is customizable by customers.

Recovery Processor:

OPENTURBO provides 3 levels of recovery for abnormal process termination,

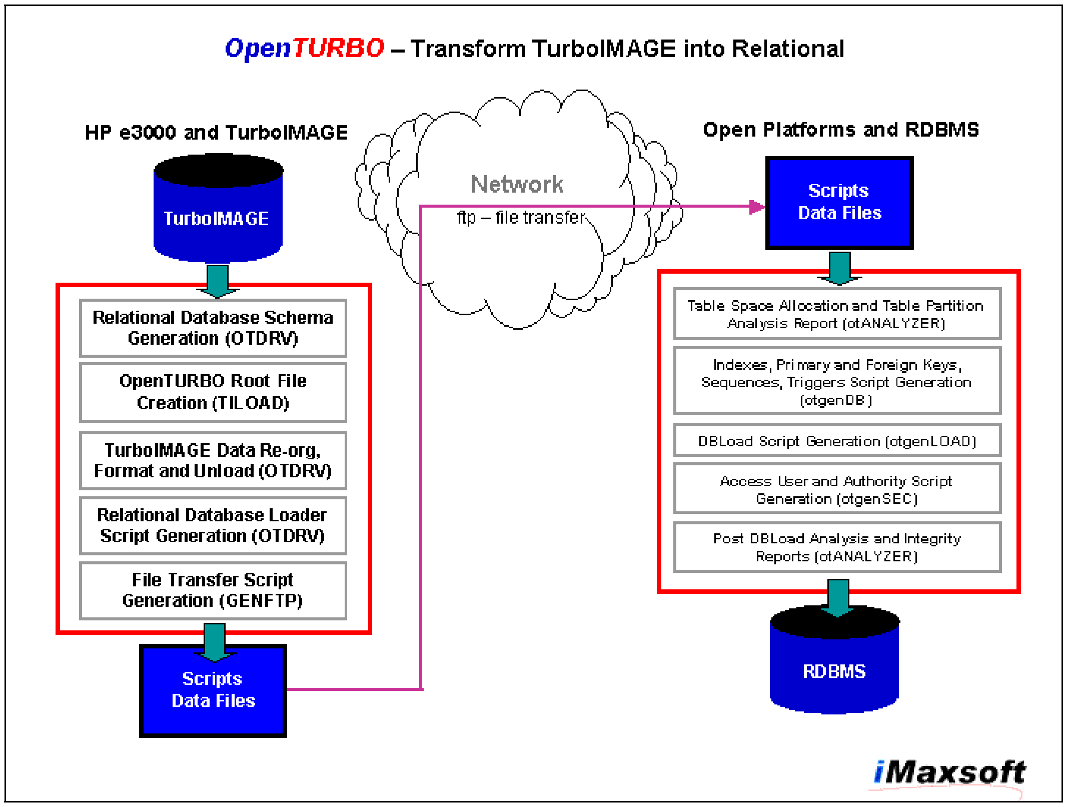
1. At client process level; any abnormal termination of the client process which is your application program, the OPENTURBO recovery process is triggered automatically to release lock objects, to close open cursors and to disconnect databases.
2. At DBSVR sever process level; any abnormal client process termination, the OPENTURBO recovery process is triggered to terminate client related dangling DBSVR processes, to re-claim all client and server processes related database control resources, and to roll-back un-committed transactions.
3. At LISTENER process level; any core abortion of DBSVR server process, the OPENTURBO recovery process is triggered to clean-up dangling server processes, to re-claim all client and server processes related database control resources, and to roll-back un-committed transactions.

OPENTURBO transaction level recovery: in a multi-database transaction, if any failure occurs during COMMIT or ROOLBACK, OPENTURBO will triggered recovery process to first undo and reset the recovery state back to the first internal COMMIT or ROLLBACK check-point, then re-try the whole COMMIT or ROLLBACK process, the number of re-tries is configurable.

OPENTURBO Utilities:

Database Migration Utility:

It transforms TurboIMAGE database into ORACLE on HP-UX or other open platforms, and maintain and manage on-going OPENTURBO database structure changes in ORACLE.


OpenTURBO – Transform TurboIMAGE into Relational
HP e3000 and TurboIMAGE
TurboIMAGE
Relational Database Schema Generation (OTDRV)
OpenTURBO Root File Creation (TILOAD)
TurboIMAGE Data Re-org, Format and Unload (OTDRV)
Relational Database Loader Script Generation (OTDRV)
File Transfer Script Generation (GENFTP)
Scripts Data Files
Network
ftp – file transfer
Open Platforms and RDBMS
Scripts Data Files
Table Space Allocation and Table Partition Analysis Report (otANALYZER)
Indexes, Primary and Foreign Keys, Sequences, Triggers Script Generation (otgenDB)
DBLoad Script Generation (otgenLOAD)
Access User and Authority Script Generation (otgenSEC)
Post DBLoad Analysis and Integrity Reports (otANALYZER)
RDBMS
iMaxsoft

ALLBASE/SQL to ORACLE or other RDBMS:

1. SQLGENX - generates ORACLE database from ALLBASE/SQL.
2. FASTLOAD - loads data from ALLBASE/SQL into ORACLE.
3. PSQLX - scans pre-processor code to identify and to flag any code change requirements.

HP e3000 Database Structure Management Utilities:

OTDRV	- Generates ORACLE schema, sets mapping and data conversion rules, unload data, format data, create loader script, transfer data via network directly or ftp, trigger loader process from HP-UX, executes data cleansing routines, and produces integrity reports.
TILOAD	- Creates OPENTURBO TIFile, verifies OPENTURBO TIFile version, re-generates TurboIMAGE schema from OPENTURBO TIFile, and mimics DBSCHEMA.PUB.SYS functionalities.
Structure Change	- You may conduct TurboIMAGE alike structure change to ORACLE on HP-UX via OPENTURBO OTDRV, TILOAD, otDBCOPY, otDBPURGE, otCLEAN, otDBUTIL, and otsyncSEQ.

HP-UX Database Structure Management Utilities:

TILOAD	- Verifies OPENTURBO TIFile version and re-generates TurboIMAGE schema from OPENTURBO TIFile.
OTDRV	- Re-generate ORACLE schema from OPENTURBO TIFile.
otCHECKChecksum	- Verify consistency between ORACLE structure versus TIFile.
otSETChecksum	- Verify and re-sync consistency between ORACLE structure and TIFile.
otgenDB	- Creates additional ORACLE specific database objects on top of standard OPENTURBO relational database creation script, such as triggers, synonym names, sequences, and etc.
otgenLOAD	- Enhances ORACLE loader script for performance, such as DIRECT=TRUE and UNRECOVERABLE for ORACLE sqlldr.
otgenSEC	- Generates ORACLE user and access securities based upon TurboIMAGE passwords and access classes.
otCLEAN	- Cleans-up Automatic table dangling entries that have no foreign constrains to any Detail tables.
otsyncSEQ	- Make sure that last-used sequence is the one being used to avoid ORACLE Unique Constraint Violation.

HP e3000 Rapid Data Unload and Transform Utilities:

OTDRV	- Unloads data, paths, and unique record identifier from TurboIMAGE datasets into flat files, generates ORACLE loader script and sets data conversion rules.
GENFTP	- Generates ftp script for transferring files from HP e3000 to HP-UX.

QUERY.PUB.SYS (Referred to TurboIMAGE/QUERY Manual) – HP e3000 only.

DBAQ (Referred to TurboIMAGE/QUERY Manual) – ORALCE on HP-UX and LINUX, a replica of QUERY.PUB.SYS.

OPENTURBO **Debugger**:

1. Programmatically:

   DBCONTROL Mode 88 – turns on or off a specific OPENTURBO debugging level. OPENTURBO supports 32 debugging levels starting from 0 through 31.

LTDBG0	- OPENTURBO Internal Core ERROR
LTDBG1	- OPENTURBO Core Library Call Trace
LTDBG2	- OPENTURBO Reserved Words
LTDBG3	- OPENTURBO Error Messages
LTDBG4	- OPENTURBO Emulator Call Trace
LTDBG5	- OPENTURBO SDK Call Trace and CURSOR POOL Size
LTDBG6	- OPENTURBO DUAL MODE Diff Results
LTDBG7	- OPENTURBO Transaction Performance Trace
LTDBG13	- TurboIMAGE Call Flow Trace
LTDBG17	– Network Traffic Dump in Hex and Text formats
LTDBG18	– Socket Information

```
    LTDBG19   – Net/IPC Information

    LTDBG27   – Dynamic SQL Statement Preparation Trace
    LTDBG28   – SQL Statement Execution Error
    LTDBG29   – MALLOC, CALLOC and FREE Tracing
```

DBCONTROL Mode 88 – set remote DBSVR debugging levels.
DBCONTROL Mode 89 – set remote DBSVR debugging output file name.

2. Setup from Client Process: the following setup is only for your-program and output to your tty.

```
EXPORT LTDBG6=1
EXPORT LTDBGOUT=-
./your-program
```

3. Setup from Server LISTENER Process: the following setup will trigger all server DBSVR processes share the same debugging output file /users/lee/tmp/2002-07-16.dbg with same debugging levels, 4, 17, 18, 19, 27, 28, and 29.

```
EXPORT LTDBG4=1
EXPORT LTDBG17=1
EXPORT LTDBG18=1
EXPORT LTDBG19=1
EXPORT LTDBG27=1
EXPORT LTDBG28=1
EXPORT LTDBG29=1
EXPORT LTDBGOUT=/users/lee/tmp/2002-07-16.dbg
./listner OTB
```

HP e3000 DUAL-MODE Differ: option to validate database migration.

Your can turn on internal DUAL-MODE option from OPENTURBO HP3000 emulator library by setting OT_DUALMODE = ON in the CONFIG file, and you must set the followings environment variables in order to view diff results:

```
SETVAR LTDBG6 1
SETVAR LTDBGOUT “difffile.group.account”
RUN yourpgm
```

HP-UX DUAL-MODE Differ: option to validate application migration.

DMDRV.PUB.IMAXSOFT is the HP3000 DUAL-MODE driver program that connects to your HP9000 programs for handling TurboIMAGE native APIs remotely.

On HP3000, you must stream the listener job first, JLISTNER.PUB.IMAXSFOT, which spawn child process DMDRV.PUB.IMAXSOFT to handle all remote TurboIMAGE calls from your HP9000 program.

JLISTNER File

```
!job listener,mgr.imaxsoft
!COMMENT
!COMMENT ***************************************************
!COMMENT *  IMAXSOFT LISTENER - for DUAL-MODE from HPUX    *
!COMMENT ***************************************************
!COMMENT
!file hosts.net.sys=hosts.pub.imaxsoft
!file services.net.sys=services.pub.imaxsoft
```

```
    !purge DMDIFF.LEE > $NULL
    !build DMDIFF.LEE;rec=-80,,f,ascii;disc=100000;msg
    !setvar ltdbg6 1
    !setvar ltdbgout "DMDIFF.LEE"
    !run listner.bin.imaxsoft;info="DBA"
    !eoj
```

On HP9000, you must set the followings in the CONFIG file:

```
    . . .

    OT_DUALMODE            = ON
    TI_DUALMODE_HOST       = 207.92.64.66
    TI_DUALMODE_SERVICE    = 32600
    TI_DUALMODE_PGM        = DMDRV.BIN.IMAXSOFT
    . . .
```

HP e3000 and HP-UX OPENTURBO **Test Driver** (TIDRV and KSAMDRV):

TIDRV is used to call TurboIMAGE API interactively without writing programs. You can create a set of API calls and test them repeatedly, you can configure TIDRV in TurboIMAGE mode only, in ORACLE mode only, or in TurboIMAGE and ORACLE dual-mode. You may direct TIDRV to get input from a file and output to a file or from/to terminal. You may turn on local client debugger to view internal OPENTURBO procedure call traces and data dump, or turn on remote server debugger to view internal DBSVR SQL statements and their respective before and after data conversion dump.

TIDRV supports it own commands along with TurboIMAGE API call commands, it is the best and most efficient tool that we use to conduct OPENTURBO internal functional, system, integration and regression tests.

TIDRV is also used for performance benchmarking, its build-in language can easily setup a test suite inter-mixed with different OPENTURBO performance optimizer options and run it for a number of pre-defined cycles for emulating OLTP as well as BATCH transactions.

KSAMDRV is the same as TIDRV but for KSAM file.

HP-UX ORACLE Integrity Analyzer (otANALYZER): produces reports for path analysis, tablespace allocation estimates and database migration validation.

OPENTURBO Listener and Transaction Process Monitor:

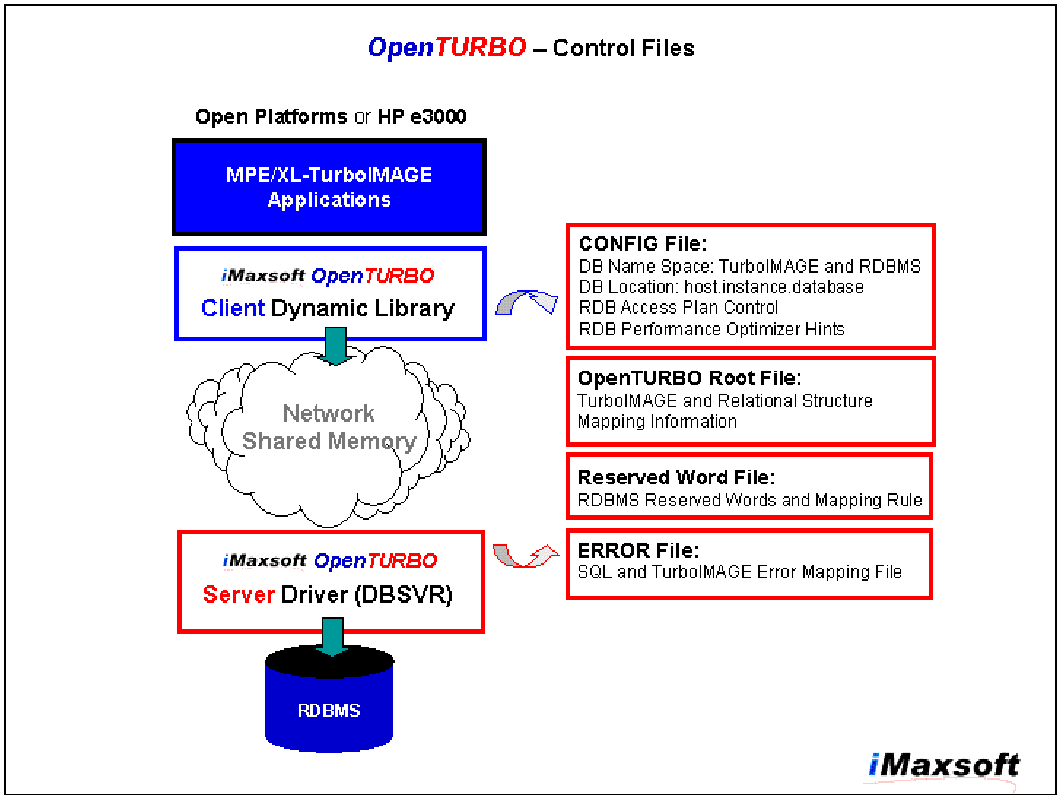
LISTNER	- OPENTURBO client-server daemon, it must be running at all time for listening your application DBOPEN connection request and spawning DBSVR server process for each corresponding client process.
LANUTIL	- Tracks listener’s activities remotely. You use it to verify who and when a specific user login and use it to gracefully shut down a listener daemon.

TurboIMAGE Remote Access Server (DMDRV):

OPENTURBO TurboIMAGE Remote Access server – DMDRV.BIN.IMAXSOFT is triggered by applications that are running on NON HP e3000 platforms and are accessing TurboIMAGE remotely via OPENTURBO. DMDRV is a persistent server, it is one HP e3000 process per corresponding DBOPEN, and it supports network byte-order transmission and NLS.

OPENTURBO Run-time Control Files:

OPENTURBO Emulator uses four major control files to resolve database names and to set database and dataset access plans (CONFIG), to get TurboIMAGE native database structure and its emulation rules (TIFILE), and to load RDBMS specific reserve words (RESERVE) and mapped messages (ERROR) for run-time usage.


OpenTURBO – Control Files
Open Platforms or HP e3000
MPE/XL-TurboIMAGE Applications
iMaxsoft OpenTURBO Client Dynamic Library
Network Shared Memory
iMaxsoft OpenTURBO Server Driver (DBSVR)
RDBMS
CONFIG File: DB Name Space: TurboIMAGE and RDBMS DB Location: host.instance.database RDB Access Plan Control RDB Performance Optimizer Hints
OpenTURBO Root File: TurboIMAGE and Relational Structure Mapping Information
Reserved Word File: RDBMS Reserved Words and Mapping Rule
ERROR File: SQL and TurboIMAGE Error Mapping File
iMaxsoft

TIFILE:	OPENTURBO Internal Root File (TIFile) contains native TurboIMAGE root file information and OPENTURBO RDBMS emulation rules. TIFile is the root of OPENTURBO technology, it is a data structure that manages and controls the entire OPENTURBO's flows.
CONFIG:	OPENTURBO CONFIG file contains rules for database name resolution (similar for FILE command), database connection method (SQLNet, persistent ORACLE server, multi-threaded ORACLE serve, etc.), database access method (trim white-space, OS platform, DBMS, etc.) , dataset access methods (bulk, tree-traverse, serial with order by, etc.), locking method (predicate only, no update lock-coverage, etc.), transaction isolation (bi-directional data replication, log-recovery, etc.), performance optimizer hint (query caching, cursor caching, bulk chain get, etc.) and etc. You can have one for your entire system, or one for each database, or even one for each application or program.
RESERVE:	ORACLE or other reserved words and replacement rule.
ERROR:	ORACLE or other to TurboIMAGE error mapping and messages.

Conclusion:

OPENTURBO is a platform that can help you quickly move and migrate your HP e3000 legacy applications to an open platform, regardless of your DBMS, TurboIMAGE, KSAM or flat files, and regardless of your development tools, 3GL and 4GL, SUPRTOOL, COGNOS, SPEEDWARE, COBOLII, VPLUS, MPEX, OMINDEX, SUPERDEX, PASCAL, FORTRAN, BBASIC, SPL, BRW, QUERY, ASKPLUS, and many more.

Our methodology is risk-free and parallelism, you must be able to migrate your applications in phases, co-existence and parallelism are embedded and transparent, and you always can roll-back your deployment for failures.

We believe automation is the key to reduce human errors during migration, and we also understand 100% automation is not possible, so we create many integrity checkers, cross-complier profiler, automated QA and Testing plug-ins, performance analyzer, converters and de-converters, and transformation standard rules book to assistant you and to assure you a successful migration.

A successful migration involves more that just code convert, it also has to provide a seamless computing environment for IT, Data Center, Network Operation, and end-users. Performance, high-availability, scalability, reliability and supportability are migration’s biggest challenges.

~ END ~