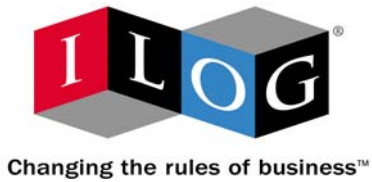

ILOG CPLEX 10.0

Release Notes

January 2006

COPYRIGHT NOTICE

General Use Restrictions

Trademarks

document version 10.0

Table of Contents

ILOG CPLEX 10.0 Release Notes

These release notes highlight improvements and new features in ILOG CPLEX 10.0. Please review these notes before using ILOG CPLEX 10.0.

Announcements

These two announcements may be of interest to you.

CPLEX Feature Changes

At times, CPLEX features are considered for redesign or removal. Before we make significant changes, we would like to understand if or how these features are being used by our customers. For the next major release, we may change or remove the following features:

- Lazy constraints
- Goals

If you have an interest in either of these features, please send an email to `cplex-productmgr@ilog.fr`.

CPLEX Updates and Future Announcements

An electronic mailing list is available to keep you informed about important product updates. If you subscribe to this list, you will receive announcements when new releases are available, updates to FAQs are published, and code samples are ready. Subscribers may possibly receive an invitation to beta testing.

To subscribe to this list, go to the ILOG Customer Support web site and navigate to the ILOG CPLEX product support pages in the Products section. The link *Subscribe to Users List* enables you access a page where you can subscribe to the ILOG CPLEX mailing list.

Only the product manager of ILOG CPLEX posts announcements to this list.

Your name and mailing address will not be published for any other purpose than receiving these official product announcements.

Conversion Notes for All Users

For users of earlier versions of ILOG CPLEX, the following topics offer guidelines for easy migration to ILOG CPLEX version 10.0. (Users of prior versions must first apply the Conversion Notes accompanying previous versions of ILOG CPLEX before upgrading to this one.)

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) see the file *yourCPLEXhome*`/mptable.html`.

Conversion notes possibly of interest to all users include these topics:

- *New Settings for Existing Parameters* on page 9

New Settings for Existing Parameters

Several new features are controlled by new values of already existing parameters. The existence of these values should not affect current applications, but for completeness Table 1 on page 10 indicates where a change has been made. Consult *New Features for All APIs* on page 24 of this document for details about what the new settings do.

Table 1 *New Settings for Existing Parameters*

Parameter in Concert Technology	Parameter in Callable Library	Parameter in Interactive Optimizer	New Settings
Cliques	CPX_PARAM_CLIQUES	mip cuts cliques 3	3 (very aggressive) for clique cut parameter
Covers	CPX_PARAM_COVERS	mip cuts cover 3	3 (very aggressive) for cover cut parameter
PreslvNd	CPX_PARAM_PRESLVND	mip strategy presolvenode	2 Probes all integer-infeasible variables at each node to find those that can be fixed
RelaxPreInd	CPX_PARAM_RELAXPREIND	preprocessing relax	default is now -1 automatic
Symmetry	CPX_PARAM_SYMMETRY	pre symmetry	(former values no and yes) -1 automatic [default] 0 off 1 moderate 2 aggressive 3 very aggressive

Read Limit Parameters: Larger Defaults

Table 2 on page 11 summarizes the parameters that set limits on pre-allocated memory in terms of the initial number of rows (constraints), columns (variables), nonzeros, and QP nonzeros to be read by default. These parameters do not limit the size of the problem that can be read. Rather, they indirectly specify the default amount of memory that will be pre-allocated for reading a problem from a file. If the limit is exceeded, more memory is allocated automatically.

The default values of these parameters have been significantly increased. Consequently, unless you have a very large problem for which fragmentation of memory has become an issue, you should not need to change these parameters from their new default values.

Table 2 *Read Limit Parameter Defaults Increased*

Concert Technology	Callable Library	Interactive Optimizer	New Default Limit
ColReadLim	CPX_PARAM_COLREADLIM	read variables	60 000
NzReadLim	CPX_PARAM_NZREADLIM	read nonzeros	250 000
QPNzReadLim	CPX_PARAM_QPNZREADLIM	read qpnonzeros	5 000
RowReadLim	CPX_PARAM_ROWREADLIM	read constraints	30 000

AdvInd Replaces MIP Start Parameter

The MIP start parameter (`MIPStart` in Concert Technology and `CPX_PARAM_MIPSTART` in the Callable Library) has been removed in favor of a more uniform interface. Use the parameter `AdvInd` in Concert Technology and `CPX_PARAM_ADVIND` in the Callable Library instead for MIP starts. In the Interactive Optimizer, the command `set mip strategy mipstart` is no longer available; use `set advance` instead.

There are also changes in the behavior of the parameter `AdvInd` in Concert Technology and `CPX_PARAM_ADVIND` in the Callable Library:

- If you do **not** want ILOG CPLEX to use a MIP start, set `AdvInd` to 0 (zero).
- If you want ILOG CPLEX to use a MIP start, leave `AdvInd` at its default value of 1 (one) or set it to 2.

Changes in MIP Start Behavior

A number of improvements have been made in how MIP starts behave. Here are a few important implications of these changes:

- MIP starts now accept partial solutions. In other words, you no longer need to supply a value for every variable in the model.
- MIP starts now accept fractional (that is, noninteger) solutions.
- If a user provides a MIP start (full or partial) that cannot be extended into a feasible solution, ILOG CPLEX will try to repair it. For more detail about this feature, see also *Repairing Infeasible MIP Starts: RepairTries* on page 36 in these release notes.
- MIP starts were formerly processed only once, at the root node; now a MIP start that you supply to ILOG CPLEX will be processed each time you invoke the MIP optimizer (`solve` in Concert Technology; `CPXmipopt` in the Callable Library; `mipopt` in the

Interactive Optimizer). For example, you could end `mipopt` after a certain number of nodes, go off and compute a solution on your own, and then give that solution to ILOG CPLEX and continue optimizing with another call to `mipopt`.

- Concert Technology offers the same functionality now for MIP starts as for continuous LP starts.

  Callable Library users now have similar ability to change parts of an existing MIP start with the new routine `CPXchgmipstart`.

  In fact, a MIP start now always starts from whatever portions of an advanced solution that ILOG CPLEX has been able to retain through model changes.

- MIP starts are now compatible with linearized constraints.
- MIP starts are now compatible with lazy constraints.
- MST files (that is, files with the extension `.mst` in MST format) can be read for an advanced start. The MST format has changed to an XML format. For more detail about that topic, see *XML Solution Files* on page 37.
- SOL files (that is, files with the extension .sol in XML format) can be read for an advanced start.
- In previous versions, to discard the search tree and start over with a MIP start, a user could set the parameters `AdvInd` to 0 (zero) and `MIPStart` to 1 (one). Now to achieve this effect, you **change the model** and set `AdvInd` to 1 (one) to begin a new search and to use whatever part of the MIP start has been retained after the change in the model. *AdvInd Replaces MIP Start Parameter* on page 11 in these release notes tells you more about this parameter.

IIS Subsumed by Conflict Refinement

The facility for finding an irreducibly inconsistent set of constraints (the IIS finder) has been subsumed by a new algorithm for finding and refining a set of conflicting constraints and bounds in an infeasible model.

See the topic *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30 of these release notes for a brief description of this new algorithm. For greater detail about this new facility, see the reference manual of your preferred API as well as the new chapter of the *ILOG CPLEX User's Manual*, *Diagnosing Infeasibility by Refining Conflicts* on page 347.

The IIS indicator parameter (`IISInd` and `CPX_PARAM_IISIND`) is no longer available. See the Concert methods `refineConflict` and the Callable Library routine `CPXrefineconflict` instead.

Error codes involving IIS in previous releases of ILOG CPLEX are deprecated and will be removed in a future release. See *Deprecated: Error Codes* on page 14 for a list of these.

Routines for manipulating IIS are deprecated and will be removed from the Callable Library in a future release. See *Deprecated: Callable Library Routines for IIS* on page 21 for a list of those deprecated routines.

Removed: Memory Management Parameters

Table 3 on page 13 summarizes parameters that have been removed in favor of a simplified and more nearly uniform interface for managing memory.

Table 3 *Parameters Removed: Memory Management*

Concert Technology	Callable Library	Interactive Optimizer
BarOOC	CPX_PARAM_BAROOC	barrier outofcore
FinalFactor	CPX_PARAM_FINALFACTOR	simplex finalfactor
PreCompress	CPX_PARAM_PRECOMPRESS	preprocessing compress

These parameters have been replaced by a single parameter to emphasize memory management, especially conservation of memory where possible. See *Memory Emphasis* on page 35.

In addition, Table 4 on page 13 summarizes growth parameters that have been removed as well. Better memory management in ILOG CPLEX and increased memory on modern platforms mean that these parameters are no longer necessary.

Table 4 *Parameters Removed: Growth Parameters*

Concert Technology	Callable Library	Interactive Optimizer
ColGrowth	CPX_PARAM_COLGROWTH	read variables
NZGrowth	CPX_PARAM_NZGROWTH	read nonzeros
QPNZGrowth	CPX_PARAM_QPNZGROWTH	read qpnonzeros
RowGrowth	CPX_PARAM_ROWGROWTH	read constraints

In contrast, the barrier growth parameter (`BarGrowth` in Concert Technology and `CPX_PARAM_BARGROWTH` in the Callable Library) is still available.

Removed: File Formats

These file formats have been removed:

- QP file format
- TRE file format

The methods or routines supporting these particular formats have also been removed. More general methods or routines are available instead. See these topics in the release notes for more detail about alternatives to these file formats:

- *XML Solution Files* on page 37
- *Change in CPXsolwrite* on page 22
- *Removed: CPXreadcopytree and CPXtreewrite* on page 20
- *Removed: CPXqpread and CPXqpwrite* on page 20

Deprecated: File Formats

The following file formats are deprecated:

- BIN (file extension `.bin`) binary representation solution files are **deprecated**
- IIS (file extension `.iis`) formatted ASCII-based representation of irreducibly inconsistent sets is **deprecated**. This format has been replaced by CLP (file extension `.clp`) for representing files in the conflict refiner.
- SOS (file extension .sos) formatted ASCII-based representation of special ordered sets (SOS) is `deprecated`.
- TXT (file extension `.txt`) formatted ASCII-based solution files are **deprecated**
- VEC (file extension `.vec`) barrier solution files are **deprecated**

See *XML Solution Files* on page 37 among New Features in these release notes for more detail about replacing these deprecated file formats and routines.

See *Deprecated: Callable Library Routines for Obsolete File Formats* on page 22 for a list of Callable Library routines supporting these deprecated file formats. These routines are also deprecated.

Deprecated: Error Codes

These error codes referring to irreducibly inconsistent sets (IIS) have been **deprecated**. They will be removed in a future release. For information about the conflict refiner (which has subsumed the IIS finder and thus made these error codes obsolete), see *IIS Subsumed by Conflict Refinement* on page 12 and *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30 in these release notes.

- `CPXERR_IIS_NO_INFO` 1701
- `CPXERR_IIS_NO_SOLN` 1702
- `CPXERR_IIS_FEAS` 1703
- `CPXERR_IIS_NOT_INFEAS` 1704

- CPXERR_IIS_OPT_INFEAS 1705
- CPXERR_IIS_DEFAULT 1706
- CPXERR_IIS_NO_BASIC 1707
- CPXERR_IIS_NO_LOAD 1709
- CPXERR_IIS_SUB_OBJ_LIM 1710
- CPXERR_IIS_SUB_IT_LIM 1711
- CPXERR_IIS_SUB_TIME_LIM 1712
- CPXERR_IIS_NUM_BEST 1713
- CPXERR_IIS_SUB_ABORT 1714

Removed: Error Codes

The MIP-specific error CPXERR_NO_INT_SOLN (3017) has been replaced by the generic error CPXERR_NO_SOLN (1217).

These error codes referring to an obsolete file format no longer exist:

- CPXERR_PAR_NO_HEADER 1525
- CPXERR_PAR_BAD_HEADER 1526
- CPXERR_PAR_SHORT 1527
- CPXERR_PAR_DATA 1528
- CPXERR_TRE_FILE_COLS 3405
- CPXERR_TRE_FILE_DATA 3401
- CPXERR_TRE_FILE_FORMAT 3414
- CPXERR_TRE_FILE_INTS 3407
- CPXERR_TRE_FILE_NONZ 3408
- CPXERR_TRE_FILE_OBJ 3404
- CPXERR_TRE_FILE_OBJSEN 3415
- CPXERR_TRE_FILE_PRESOLVE 3410
- CPXERR_TRE_FILE_ROWS 3406
- CPXERR_TRE_FILE_TYPES 3409
- CPXERR_TRE_FILE_VERSION 3403
- CPXERR_TRE_FILE_WRITE 3402

The conditions that could raise those error codes no longer occur.

Deprecated: ReverseInd Parameter

The parameter for specifying that the byte order should be reversed when ILOG CPLEX reads SAV files is **deprecated** in this release. It will be removed in a future release. There is no longer need for you to set this parameter manually, as automatic facilities in ILOG CPLEX now detect the byte order of data and behave appropriately. The name of this deprecated parameter in Concert Technology is `ReverseInd`. In the Callable Library, it is `CPX_PARAM_REVERSEIND`. In the Interactive Optimizer, the deprecated command is `set read reverse`.

Deprecated: XXXInd and BasInterval Parameters

The parameter to enable and disable the reading of XXX files has been **deprecated** and will be removed in a future release. Likewise, the parameter for setting the frequency for writing a basis to a file has been deprecated and will be removed in a future release.

Table 5 *Parameters Deprecated: XXX File Indicator and BasInterval*

Concert Technology	Callable Library	Interactive Optimizer
`XXXInd`	`CPX_PARAM_XXXIND`	`simplex xxxstart`
`BasInterval`	`CPX_PARAM_BASINTERVAL`	`simplex basisinterval`

Conversion Notes for Microsoft Users

Please note that the standard distribution of ILOG CPLEX 10.0 contains these subdirectories or folders of interest to Microsoft users:

- *yourCPLEXhome*`\lib\x86_.net2003_7.1\stat*`
- *yourCPLEXhome*`\lib\ia64_.net2003_7.1\stat*`
- *yourCPLEXhome*`\lib\x64_.net2005_8.0\sta*`
- *yourCPLEXhome*`\bin\x86_win32`
- *yourCPLEXhome*`\bin\ia64_win64`
- *yourCPLEXhome*`\bin\x64_win64`

where *`yourCPLEXhome`* indicates the place on your Microsoft platform where you installed the product, and `bin` and `lib` are distinct folders there.

Documentation for `CPLEX.NET` is available in compiled Microsoft Help format (CHM) as part of the ILOG Optimization Documentation suite.

Conversion Notes for Concert Technology Users

These changes in Concert Technology may affect existing ILOG CPLEX applications:

- Constructors of callbacks now take an instance of `IloEnv` as an argument.
- The signature of the method `feasOpt` has changed. The final argument (the optimization indicator) has been removed. A new Concert Technology parameter, `FeasOptMode`, has been introduced. For more detail, see *FeasOpt Relaxation Modes* on page 26 among the new features in these release notes.

  Furthermore, the behavior has changed with respect to what it returns. For more detail about this change, see *After FeasOpt* on page 29 in these release notes.

  These changes in the signature and return behavior of `feasOpt` mean that if your application uses this method, you will need to edit and recompile.
- The Java class `IloCplex` now derives from a new class, `IloCplexModeler`, which implements the familiar interface `IloMPModeler`. All the modeling methods previously in `IloCplex` are now available in `IloCplexModeler`. (The solution methods remain in `IloCplex`.) This re-organization makes it possible for a user to build models in a Java application as pure Java objects, without using the class `IloCplex`.

  In particular, a model built with `IloCplexModeler` using no instance of `IloCplex` does not require loading of the `CPLEX.dll` nor any shared library.

  Furthermore, `IloCplexModeler` is serializable. For example, a user may develop a pure Java application that builds a model with `IloCplexModeler` and sends the model and modeling objects off to an optimization server that uses `IloCplex`. The example `CplexServer.java` shows you how to write such an application.

  The class `IloCplex` extends `IloCplexModeler`, so existing applications will continue to run correctly without change.
- New quality type: `ExactKappa`. You can access exact kappa by means of the method:
  - C++ API: `IloCplex::getQuality`
  - Java API: `IloCplex.getQuality`
  - .NET API: `Cplex.GetQuality`

Deprecated methods will be removed from future releases.

- C++ API deprecated member functions
  - `IloCplex::getNcliques` **deprecated**

    See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.
  - `IloCplex::getNcovers` **deprecated**

See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

- `IloCplex::getFormulationEpsValue` **deprecated**

  Use the parameter `IloCplex::EpLin` instead.

- `IloCplex::readVectors` **deprecated**
- `IloCplex::writeVectors` **deprecated**
- `IloCplex::writeTextSolution` **deprecated**
- `IloCplex::getIIS` **deprecated**

◆ Java API deprecated methods

- `IloCplex.getNcliques` **deprecated**

  See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

- `IloCplex.getNcovers` **deprecated**

  See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

- `IloCplex.readVectors` **deprecated**
- `IloCplex.writeVectors` **deprecated**
- `IloCplex.writeTextSolution` **deprecated**
- `IloCplex.getIIS` **deprecated**

◆ .NET API deprecated methods

- `Cplex.Ncliques` **deprecated**

  See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

- `Cplex.Ncovers` **deprecated**

  See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

- `Cplex.ReadVectors` **deprecated**
- `Cplex.WriteVectors` **deprecated**
- `Cplex.WriteTextSolution` **deprecated**
- `Cplex.GetIIS` **deprecated**

Removed methods

- These member functions have been removed from the C++ API:
  - `IloCplex::getFormulationBigValue` **removed**
  - `IloCplex::setFormulationBigValue` **removed**
  - `IloCplex::getPWLFormulation` **removed**
  - `IloCplex::setPWLFormulation` and the enumeration `IloCplex::PWLFormulation` **removed**
  - `IloCplex::readTree` **removed**
  - `IloCplex::writeTree` **removed**
- These methods have been removed from the Java API:
  - `IloCplex.readTree` **removed**
  - `IloCplex.writeTree` **removed**
- These methods have been removed from the .NET API:
  - `Cplex.ReadTree` **removed**
  - `Cplex.WriteTree` **removed**

Conversion Notes for Callable Library Users

These changes in the Callable Library may affect existing ILOG CPLEX applications:

Change in CPXfeasopt

`CPXfeasopt` no longer includes the optind argument to determine whether CPLEX should find the optimal solution among all minimal relaxations. Instead, use the new parameter governing the mode of FeasOpt, described in *FeasOpt Relaxation Modes* on page 26 of these release notes. This change in the signature of this routine means that if your application uses it, you will need to edit your application and recompile.

Removed: CPXreadcopytree and CPXtreewrite

The routines `CPXreadcopytree` and `CPXtreewrite`, which supported the TRE file format deprecated in ILOG CPLEX version 9.1, have been removed.

For advanced starts, enter a full or partial solution by means of the routine `CPXcopymipstart`.

For solutions, use the new routine `CPXreadcopysol` to read SOL files in XML format. See *XML Solution Files* on page 37 among New Features in these release notes for more about solution files.

Removed: CPXqpread and CPXqpwrite

The routines `CPXqpread` for reading and `CPXqpwrite` for writing a QP format file corresponding to the quadratic coefficient matrix of a quadratic program were deprecated in a previous release, and they have been removed in this release. QP-specific files are no longer necessary. Use the more generic read and write routines `CPXreadcopyprob` and `CPXwriteprob` instead.

Deprecated: Accessing Solution Quality

In the Callable Library, the routine `CPXgetdblquality` now provides a uniform API for accessing solution quality. The following advanced routines, formerly needed for accessing solution quality, have been deprecated and will be removed from future releases:

- `CPXcheckpib` **deprecated**
- `CPXcheckax` **deprecated**
- `CPXgetExactkappa` **deprecated**
- `CPXgetkappa` **deprecated**

Use `CPXgetdblquality` with appropriate arguments instead.

A new type of quality (`CPX_EXACT_KAPPA` in the Callable Library and `ExactKappa` in Concert Technology) has been added to support this uniform API for accessing solution quality by means of `CPXgetdblquality`.

Deprecated: MIP-Specific Routines

These MIP-specific routines of the Callable Library are now deprecated and will be removed from future releases:

- `CPXgetmipx` **deprecated**
- `CPXgetmipslack` **deprecated**
- `CPXgetmipobjval` **deprecated**
- `CPXgetmipqconstrslack` **deprecated**

Use these generic routines instead:

- `CPXgetx`
- `CPXgetslack`
- `CPXgetobjval`
- `CPXgetqconstrslack`

Deprecated: Callable Library Routines for IIS

The IIS finder, a feature for calculating irreducibly inconsistent sets, has been subsumed by the conflict refiner, introduced in these release notes in *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30. As a consequence of extending the IIS finder to this more generic conflict refiner, the following routines of the Callable Library are deprecated and will be removed from a future release:

- `CPXdisplayiis` **deprecated**
- `CPXfindiis` **deprecated**
- `CPXgetiis` **deprecated**
- `CPXiiswrite` **deprecated**

Deprecated: Callable Library Routines for Counting Clique and Cover Cuts

Routines in the Callable Library for counting clique cuts and cover cuts have been deprecated:

- `CPXgetclqcnt` **deprecated**
- `CPXgetcovcnt` **deprecated**
- `CPXgetgenclqcnt` **deprecated**

See *Counting Cuts* on page 259 in the *ILOG CPLEX User's Manual*, for alternative ways of counting cuts.

Deprecated: Callable Library Routines for Obsolete File Formats

These routines of the Callable Library are deprecated and will be removed from future releases:

- `CPXbinsolwrite` **deprecated**
- `CPXreadcopyvec` **deprecated**
- `CPXtxtsolwrite` **deprecated**
- `CPXwritesol` **deprecated**
- `CPXwritevec` **deprecated**

Those routines read or write file formats that have been deprecated. Instead, use the new Callable Library routine `CPXreadcopysol`, which provides more generic access to a wider variety of file formats, and the changed routine `CPXsolwrite`, which accesses the new XML solution format in files with the file extension `.sol` written in XML syntax. For more detail about XML format in ILOG CPLEX, see *XML Solution Files* on page 37 in these release notes.

Also SOS (file extension `.sos`) formatted ASCII-based files for declaring special ordered sets (SOS) are **deprecated**. The Callable Library routines for reading and writing files in this format are also deprecated and will be removed from future releases:

- `CPXreadcopysos` **deprecated**
- `CPXsoswrite` **deprecated**

Change in CPXsolwrite

The routine `CPXsolwrite` has changed. Its purpose now is to write a file in the new XML solution format. See *XML Solution Files* on page 37 in these release notes for more information about this change. The routine is no longer an advanced routine.

Changes for SOS

Two routines of the Callable Library that you use to specify special ordered sets (SOSs), `CPXcopysos` and `CPXaddsos`, have changed. The argument to specify branching priority for each SOS has been removed from both routines. An argument to specify the names of the SOSs has been added.

The routine of the Callable Library for accessing a special ordered set `CPXgetsos` has also changed. The argument `sospri` to specify priority has been removed.

The symbolic value `CPX_CALLBACK_INFO_SOS_PRIORITY` has been removed. This symbol for specifying priority of a special ordered set (SOS) was formerly a possible

argument of `CPXgetcallbackinfo`, the routine of the Callable Library for querying callback information. This symbol is no longer an argument for that routine.

These changes in the signatures of these routines mean that if your application uses them, you must edit your application and recompile.

These changes also mean that when the names of SOSs appear in LP files (that is, formatted files with the extension `.lp`) or MPS files (that is, formatted files with the extension `.mps`), the names will be retained and used by ILOG CPLEX. The names will no longer be discarded, as they were in the past.

Furthermore, output files in the formats LP, SAV, and MPS will now show any names that the user assigned or that were read from input files.

New routines `CPXgetsosname` and `CPXgetsosindex` have been added to the Callable Library. `CPXgetsosname` accesses a range of names of special ordered sets in a CPLEX problem object. `CPXgetsosindex` searches for the index of a special ordered set in a CPLEX problem object.

Names in User-Defined Cuts and in Lazy Constraints

Two **advanced** routines of the Callable Library now take an additional argument that allows a user to provide a name for a user-defined cut or a lazy constraint in the C API:

- `CPXaddusercuts` takes an additional argument to indicate the name of the cuts. The additional argument is an array of names; it may be NULL.
- `CPXaddlazyconstraints` takes an additional argument to indicate the name of a lazy constraint. Again, the additional argument is an array of names; it may be NULL.

This change in the signature of these **advanced** routines means that if your application uses them, you will need to edit your application and recompile.

This change also means that when names of cuts or lazy constraints appear in LP files (that is, formatted files with the extension `.lp`), the names will be retained and used; the names will no longer be discarded, as they were before.

Furthermore, output files in the formats LP, SAV, and MPS will now show any names that the user assigned or that were read from input files.

In fact, the MPS format has been changed to accommodate lazy constraints and user-defined cuts. You can now write lazy constraints and user-defined cuts to MPS files and read lazy constraints and user-defined cuts from MPS files.

Conversion Notes for Interactive Optimizer Users

In addition to the New Features detailed in *New Features in the Interactive Optimizer* on page 43 in these release notes, the following changes may be of immediate interest to users of the Interactive Optimizer.

- The former command `set mip emphasis` has changed its syntax. It is now:

  ```
  set emphasis mip
  ```

  This change makes the emphasis command consistent for emphasis on MIP features, numeric caution, and memory use:

  - `set emphasis mip`
  - `set emphasis numeric`
  - `set emphasis memory`

- The command to display an irreducibly inconsistent set (`display iis`) has been removed; instead, use the command `conflict` followed by the command `display conflict`.

  For more about the new feature, the conflict refiner, which has subsumed the IIS finder, see *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30 among the New Features in these release notes.

New Features for All APIs

Here's more detail about major new features in this release:

- *New Parameters*
- *FeasOpt Relaxation Modes* on page 26
- *After FeasOpt* on page 29
- *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30
- *Indicator Constraints* on page 32
- *Improvements for MIQPs* on page 33
- *Improvements in Presolve: RepeatPresolve* on page 33
- *Cliques* on page 33
- *Covers* on page 33
- *PreslvNd* on page 34

- *FeasOpt Available in Interactive Optimizer* on page 43
- *Infeasibility as Conflict in the Interactive Optimizer* on page 44
- *Longer Variable Names in Interactive Optimizer* on page 45
- *Greater Precision in MPS, REW Files in Interactive Optimizer* on page 45
- *Wildcards in Interactive Optimizer* on page 45

FeasOpt Relaxation Modes

FeasOpt, introduced in a previous version of ILOG CPLEX, performs an additional optimization that computes a minimal relaxation of the constraints and bounds of variables and righthand sides of constraints in an infeasible model to make the active model feasible. In this version, a new parameter has been introduced to guide FeasOpt in its computation of that relaxation.

- `FeasOptMode` in Concert Technology
- `CPX_PARAM_FEASOPTMODE` in the Callable Library
- `set feasopt` *`mode`* in the Interactive Optimizer

FeasOpt works in two phases. In its first phase, it attempts to minimize its relaxation of the infeasible model. That is, it attempts to find a feasible solution that requires minimal change. In its second phase, it finds an optimal solution with respect to the original objective among those that require only as much relaxation as it found necessary in the first phase.

Your choice of values for the FeasOpt relaxation mode parameter (listed in Table 6 on page 27 in these release notes) indicates two aspects to ILOG CPLEX:

- whether to stop in phase one or continue to phase two:
  - Min means stop in phase one with a minimal relaxation.
  - Opt means continue to phase two for an optimum among those minimal relaxations.
- how to measure the minimality of the relaxation:
  - Sum means ILOG CPLEX should minimize the sum of all relaxations
  - Inf means that ILOG CPLEX should minimize the number of constraints and bounds relaxed.
  - Quad means that ILOG CPLEX should minimize the sum of the squares of the relaxations.

Table 6 *New Relaxation Modes for FeasOpt*

Concert	Callable Library	Interactive Optimizer	Effect
`MinSum`	`CPX_FEASOPT_MIN_SUM`	0	Minimize the sum of all required relaxations in first phase only
`OptSum`	`CPX_FEASOPT_OPT_SUM`	1	Minimize the sum of all required relaxations in first phase and execute second phase to find optimum with respect to the original objective among minimal relaxations
`MinInf`	`CPX_FEASOPT_MIN_INF`	2	Minimize the number of constraints and bounds requiring relaxation in first phase only
`OptInf`	`CPX_FEASOPT_OPT_INF`	3	Minimize the number of constraints and bounds requiring relaxation in first phase and execute second phase to find optimum with respect to the original objective among minimal relaxations
`MinQuad`	`CPX_FEASOPT_MIN_QUAD`	4	Minimize the sum of squares of required relaxations in first phase only
`OptQuad`	`CPX_FEASOPT_OPT_QUAD`	5	Minimize the sum of squares of required relaxations in first phase and execute second phase to find optimum among minimal relaxations

Existing status values for relaxations have been extended to include FeasOpt results. They are summarized in *FeasOpt Status* on page 28. To access these status values, use the conventional methods or routines for accessing solution status:

- In Concert Technology, use the method `getCplexStatus`
- In the Callable Library
  - For MIPs, use the routine `CPXgetstat`
  - For LPs and QPs, use the routine `CPXsolinfo`

Table 7 *FeasOpt Status*

Concert Technology	Callable Library		Meaning
FeasibleRelaxedSum	CPX_STAT_FEASIBLE_RELAXED_SUM	14	A solution was found for the model relaxed according to the criterion of minimizing the sum of relaxations; the solution is feasible.
OptimalRelaxedSum	CPX_STAT_OPTIMAL_RELAXED_SUM	15	A solution was found for the model relaxed according to the criterion of minimizing the sum of relaxations; the solution is optimal.
FeasibleRelaxedInf	CPX_STAT_FEASIBLE_RELAXED_INF	16	A solution was found for the model relaxed according to the criterion of relaxing the minimal number of constraints and bounds; the solution is feasible.
OptimalRelaxedInf	CPX_STAT_OPTIMAL_RELAXED_INF	17	A solution was found for the model relaxed according to the criterion of relaxing the minimal number of constraints and bounds; the solution is optimal.
FeasibleRelaxedQuad	CPX_STAT_FEASIBLE_RELAXED_QUAD	18	A solution was found for the model relaxed according to the criterion of minimizing the sum of squares of relaxations; the solution is feasible.
OptimalRelaxedQuad	CPX_STAT_OPTIMAL_RELAXED_QUAD	19	A solution was found for the model relaxed according to the criterion of minimizing the sum of squares of relaxations; the solution is optimal.
FeasibleRelaxedSum	CPXMIP_FEASIBLE_RELAXED_SUM	120	A solution was found for the model relaxed according to the criterion of minimizing the sum of relaxations; the solution is feasible.

Table 7 *FeasOpt Status*

Concert Technology	Callable Library		Meaning
`OptimalRelaxedSum`	`CPXMIP_OPTIMAL_RELAXED_SUM`	121	A solution was found for the model relaxed according to the criterion of minimizing the sum of relaxations; the solution is optimal.
`FeasibleRelaxedInf`	`CPXMIP_FEASIBLE_RELAXED_INF`	122	A solution was found for the model relaxed according to the criterion of relaxing the minimal number of constraints and bounds; the solution is feasible.
`OptimalRelaxedInf`	`CPXMIP_OPTIMAL_RELAXED_INF`	123	A solution was found for the model relaxed according to the criterion of relaxing the minimal number of constraints and bounds; the solution is optimal.
`FeasibleRelaxedQuad`	`CPXMIP_FEASIBLE_RELAXED_QUAD`	124	A solution was found for the model relaxed according to the criterion of minimizing the sum of squares; the solution is feasible.
`OptimalRelaxedQuad`	`CPXMIP_OPTIMAL_RELAXED_QUAD`	125	A solution was found for the model relaxed according to the criterion of minimizing the sum of squares; the solution is optimal.
`AbortRelaxed`	`CPXMIP_ABORT_RELAXED`	126	Terminated prematurely (for example, because of a time limit)

After FeasOpt

FeasOpt (the method `feasOpt` in Concert Technology and the routine `CPXfeasopt` in the Callable Library) formerly used the same arguments for two different purposes:

- as input from the user to specify preferences for relaxing bounds and constraints;
- as output to return the optimal relaxation quantities for these bounds and constraints.

FeasOpt no longer returns the relaxation quantities in this way. Instead, you can compute the extent of the optimal relaxation from the solution vector returned by FeasOpt.

In order to see which constraints and bounds FeasOpt relaxed, query the solution. In an application of the Callable Library, use these routines:

- `CPXgetrowinfeas` for rows
- `CPXgetcolinfeas` for columns
- `CPXgetqconstrinfeas` for quadratic constraints
- `CPXgetindconstrinfeas` for indicator constraints
- `CPXgetsosinfeas` for special ordered sets

To query the solution in an application of Concert Technology, use one of these methods:

In the C++ API:

- `IloCplex::getInfeasibility`
- `IloCplex::getInfeasibilities`

In the Java API:

- `IloCplex.getInfeasibility`
- `IlloCplex.getInfeasibilities`

In the .NET API:

- `Cplex.GetInfeasibility`
- `Cplex.GetInfeasibilities`

Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models

Given an infeasible model, whether LP, MIP, QP, or QCP, ILOG CPLEX can identify conflicting constraints and bounds within the model to help a user identify the causes of the infeasibility. In this context, a *conflict* is a subset of the constraints and bounds of the model which are mutually contradictory. ILOG CPLEX first examines the full infeasible model to identify portions of the conflict that it can remove. By this process of refinement, ILOG CPLEX arrives at a *minimal conflict*. A minimal conflict is usually smaller than the full infeasible model and thus makes infeasibility analysis easier. To invoke this new facility, the conflict refiner:

- In Concert Technology, use the methods:
  - `IloCplex::refineConflict` and `getConflict` in the C++ API
  - `IloCplex.refineConflict` and `getConflict` in the Java API
  - `Cplex.RefineConflict` and `GetConflict` in the .NET API
- In the Callable Library, use the routine `CPXrefineconflict`. See also the routines `CPXrefineconflictext`, `CPXgetconflict`, `CPXgetconflictext`.

- In the Interactive Optimizer, use the command `conflict`.

To write the results from the conflict refiner to a file with the file extension `.clp`, use the new method `writeConflict` in a Concert Technology application or `CPXclpwrite` in a Callable Library application.

A new parameter (`ConflictDisplay` in Concert Technology and `CPX_PARAM_CONFLICTDISPLAY` in the Callable Library) controls how much information is displayed by the conflict refiner. By default, display is on and invokes summary information. Possible values are these:

- 0 off display no information from the conflict refiner
- 1 on display summary information
- 2 on display information about all models being solved

If a model happens to include multiple independent causes of infeasibility, then it may be necessary for the user to repair one such cause and then repeat the diagnosis with further conflict analysis.

A conflict does **not** provide information about the magnitude of change in data values needed to achieve feasibility. The techniques that ILOG CPLEX uses to refine a conflict include or remove constraints or bounds in trial conflicts; the techniques do not vary the data in constraints nor in bounds. To gain insight about changes in bounds on variables and constraints, consider the FeasOpt feature. Also consider FeasOpt for an approach to automatic repair of infeasibility.

Refining a conflict in an infeasible model as defined here is similar to finding an irreducibly inconsistent set (IIS), an established technique in the published literature, long available within ILOG CPLEX. Both tools (conflict refiner and IIS finder) attempt to identify an infeasible subproblem in an infeasible model. However, the conflict refiner is more general than the IIS finder. The IIS finder is applicable only in continuous (that is, LP) models, whereas the conflict refiner can work on any type of problem, even those containing discrete variables or quadratic elements.

Also the conflict refiner differs from the IIS finder in that a user may organize constraints into one or more *groups* for a conflict. When a user specifies a group, ILOG CPLEX will make sure that either the group as a whole will be present in a conflict (that is, all its members will participate in the conflict, and removal of one will result in a feasible subproblem) or that the group will not participate in the conflict at all. See the Concert Technology methods `refineConflict` and the Callable Library routine `CPXrefineconflictext` for more about groups.

A user may also assign a numeric *preference* to constraints or to groups of constraints. In the case of an infeasible model having more than one possible conflict, preferences guide the conflict refiner toward identifying constraints in a conflict as the user prefers.

In these respects, the conflict refiner represents an extension and generalization of the IIS finder.

For examples of the conflict refiner in use, see *Diagnosing Infeasibility by Refining Conflicts* on page 347 in the *ILOG CPLEX User's Manual*.

Indicator Constraints

An *indicator constraint* is a new way of expressing relationships among variables by specifying a binary variable to control whether or not a linear constraint takes effect. For example, indicator constraints are useful in problems where there are fixed charges to express only if a given variable comes into play.

So-called Big M formulations often exhibit trickle flow, and at times, they behave unstably. The main purpose of indicator constraints is to avoid the unwanted side-effects of Big M formulations. Generally, the use of indicator constraints is **not** warranted when the unwanted side-effects of Big M formulations are not present.

In Concert Technology applications, indicator constraints are automatically invoked by ILOG CPLEX when a user declares expressions that can be linearized (such expressions as `IloIfThen`, for example).

In Callable Library applications, the new routine `CPXaddindconstr` is available for you to add indicator constraints to your model. For an example of its use, see `fixnet.c` in the examples directory of the product. To remove an indicator constraint that you have added, use the routine `CPXdelindconstr`. To access slack values of an indicator constraint, use the new routine `CPXgetindconstrslack`. To access information about the indicator constraints of a presolved model from callbacks during MIP optimization, use the new **advanced** routine `CPXgetcallbackindicatorinfo`.

In the Interactive Optimizer, you can include indicator constraints among the usual linear constraints in LP format.

For example, instead of the following Big M formulation, which relies on the x values summing to less than one billion (a formulation that can cause numeric instability or undesirable solutions in some situations):

```
constr01: x1 + x2 + x3 - 1e+9 y <= 0 \\ may cause problems
```

you can enter the following formulation, where y is a binary variable, using an indicator constraint, like this:

```
constr01: y = 0 -> x1 + x2 + x3 = 0 \\ alternative
```

Improvements for MIQPs

ILOG CPLEX version 9.1 extended its facilities for generating cuts automatically to attempt generation of Gomory cuts for mixed integer programs with a quadratic objective function (MIQP).

ILOG CPLEX version 10.0 rejects fewer models that are not positive semi-definite.

Improvements in Presolve: RepeatPresolve

A new parameter lets you tell ILOG CPLEX to re-apply the MIP presolve techniques of the preprocessor to a MIP model at the root after preprocessing has otherwise finished (that is, after cut generation at the root). The name of the parameter is:

- `RepeatPresolve` in Concert Technology
- `CPX_PARAM_REPEATPRESOLVE` in the Callable Library
- `preprocessing repeatpresolve` in the Interactive Optimizer

Settings of this new parameter are:

- -1 Automatic: Let CPLEX choose whether to re-apply presolve; this is the default setting.
- 0 Turn off repeat presolve.
- 1 Repeat presolve without cuts.
- 2 Repeat presolve with cuts.
- 3 Repeat presolve with cuts and allow new root cuts.

Cliques

A very aggressive new setting (3) is available for the clique cut parameter. The parameter name is:

- `Cliques` in Concert Technology
- `CPX_PARAM_CLIQUES` in the Callable Library
- `mip cuts cliques 3` in the Interactive Optimizer

Covers

A very aggressive new setting (3) is available for the cover cut parameter. The parameter name is:

- `Covers` in Concert Technology

- `CPX_PARAM_COVERS` in the Callable Library
- `mip cuts covers 3` in the Interactive Optimizer

PreslvNd

A new setting (2) for the presolve node parameter is available. The new setting probes all integer-infeasible variables at each node to find those that can be fixed. The name of the parameter is:

- `PreslvNd` in Concert Technology
- `CPX_PARAM_PRESLVND` in the Callable Library
- `mip strategy presolvenode 2` in the Interactive Optimizer

RelaxPreInd

The default value of the indicator for preprocessing relaxation has changed. The default is now -1 automatic: let CPLEX decide. The new default usually produces a faster initial solution of the relaxation. The name of the parameter is:

- `RelaxPreInd` in Concert Technology
- `CPX_PARAM_RELAXPREIND` in the Callable Library
- `preprocessing relax` in the Interactive Optimizer

Symmetry

ILOG CPLEX now offers more aggressive techniques for symmetry breaking. You can turn off symmetry breaking or revert to previous behavior by means of new settings in the symmetry parameter. The name of the parameter is:

- `Symmetry` in Concert Technology
- `CPX_PARAM_SYMMETRY` in the Callable Library
- `set presolve symmetry` in the Interactive Optimizer

The previous values of this parameter were binary: on or off.

The new values are:

- -1 automatic: let CPLEX decide the level of symmetry breaking; this setting is now the default.
- 0 turns off symmetry breaking
- 1 moderate - invokes behavior consistent with versions prior to 9.1
- 2 aggressive - strengthens previous behavior

- 3 very aggressive - new techniques

Epsilon Linearity: EpLin

A new parameter sets the epsilon used in linearization. The name of the parameter is `EpLin` in Concert Technology. (This parameter is not applicable in the Callable Library, nor in the Interactive Optimizer.) Its default value is `1e-3`. It controls how strict inequalities are managed during linearization. In other words, it provides an epsilon for determining when two values are not equal during linearization. For example, when `x` is a numeric variable (that is, an instance of `IloNumVar`), `x < a` becomes `x <= a-eplin`. Similarly, `x!=a` becomes `{(x < a) || (x > a)}` which is linearized as

`{( x <= a-eplin) || (x >= a+eplin)}`.

When should you change this parameter? If you are not getting an expected solution for a Concert Technology model that uses linearization, it might be that this solution is cut off because of the relatively high `EpLin` value. In such a case, carefully try reducing it. Care must be taken in reducing `EpLin`, because the smaller the epsilon, the more numerically unstable the model will tend to become.

Memory Emphasis

A new parameter lets you tell ILOG CPLEX to emphasize reduced use of memory. This new parameter replaces certain parameters as noted in *Removed: Memory Management Parameters* on page 13. Also, while solution information will be available after optimization, certain computations that require a basis that has been factored (for example, for the computation of the condition number Kappa) may be unavailable, depending on the operations performed during preprocessing.

- `MemoryEmphasis` in Concert Technology
- `CPX_PARAM_MEMORYEMPHASIS` in the Callable Library
- `emphasis memory` in the Interactive Optimizer

The settings of the memory emphasis parameter are:

- 0 `CPX_OFF` (off) the default: do not emphasize conservation of memory
- 1 `CPX_ON` (on) emphasize conservation of memory

Numerical Emphasis

A new parameter lets you tell ILOG CPLEX to exercise extreme caution with respect to numeric computations.

- `NumericalEmphasis` in Concert Technology
- `CPX_PARAM_NUMERICALEMPHASIS` in the Callable Library

- `emphasis numerical` in the Interactive Optimizer

The settings of the numerical emphasis parameter are:

- 0 `CPX_OFF` (off) the default: do not emphasize extreme caution in computations
- 1 `CPX_ON` (on) emphasize extreme caution in computations

Polishing Time

A new parameter enables you to regulate the amount of time that ILOG CPLEX spends in polishing the best solution found. During *Solution Polishing*, ILOG CPLEX applies its effort to trying to improve the best feasible solution. Polishing can yield better solutions in some situations. The default value of the polishing time parameter is 0 (zero); that is, spend no time polishing. The parameter accepts any nonnegative value to set a limit in seconds.

- `PolishTime` in Concert Technology
- `CPX_PARAM_POLISHTIME` in the Callable Library
- `mip limits polishtime` in the Interactive Optimizer

Solution Polishing can be used to improve the best known solution at the end of branch & cut if optimality has not been proven. Alternatively, it can used instead of the branch & cut algorithm if an initial solution can be found at the root node. If Solution Polishing is used as an alternative algorithm to branch & cut, optimality may not be proven even if the optimal solution is found.

ProbeTime

A new parameter enables you to limit the amount of time that ILOG CPLEX spends in probing. The default value is `1e+75` seconds. The parameter accepts any nonnegative value.

- `ProbeTime` in Concert Technology
- `CPX_PARAM_PROBETIME` in the Callable Library
- `mip limits probetime` in the Interactive Optimizer

Repairing Infeasible MIP Starts: RepairTries

If a user provides a MIP start (full or partial) that cannot be extended into a feasible solution, ILOG CPLEX will try to repair it. By default, ILOG CPLEX will try the repair heuristic once. The user can change the number of repair attempts (to any nonnegative value) by setting the parameter `RepairTries` in Concert Technology or `CPX_PARAM_REPAIRTRIES` in the Callable Library.

Precision of Output in MPS and REW Files: MPSLongNum

A new parameter lets you control the degree of precision displayed in output files in MPS and REW formats:

- `MPSLongNum` in Concert Technology
- `CPX_PARAM_MPSLONGNUM` in the Callable Library
- `output mpslong` in the Interactive Optimizer

When this parameter is set to its default value 1 (one), numbers are written to MPS files in full-precision; that is, up to 15 significant digits may be written. The setting 0 (zero) writes files that correspond to the standard MPS format, where at most 12 characters can be used to represent a value. This limit may result in loss of precision.

For more detail about the MPS and REW file formats generally, see the *ILOG CPLEX File Format Reference Manual*.

Reading Compressed Files

The routines for reading compressed files have been improved to recognize the following file types:

- `.bz2` for files compressed with BZip2 (new)
- `.Z` or `.z` for files compressed with Zip
- `.gz` for files compressed with GNU Zip

XML Solution Files

ILOG CPLEX enables you to read and write a solution file, formatted in XML, for all problem types in all APIs. These new solution files, known as SOL files, carry the file extension `.sol`. The XML solution format enables you to display these solution files in most browsers. ILOG CPLEX also offers a stylesheet and schema in the `include` directory of the product to facilitate use of this format in your applications.

- `solution.xsl` stylesheet
- `solution.xsd` schema

These generic solution files, suitable for all problem types, replace the TXT (file extension `.txt`) and BIN (file extension `.bin`) files that were formerly available only for LP problems.

ILOG CPLEX can also read these new SOL files as an advanced start. Consequently, they replace VEC (file extension `.vec`) files as well.

ILOG CPLEX can also read these new SOL files as an advanced basis. The BAS file format continues to exist, but you may want to use a SOL file in situations where you have used a BAS file before.

The MST file format (file extension `.mst`) for advanced starts has changed to this new SOL XML format.

- In Concert Technology, there are specific methods for reading and writing SOL files:
  - In the C++ API, see the methods `IloCplex::readSolution` and `IloCplex::writeSolution`.
  - In the Java API, see the methods `IloCplex.readSolution` and `IloCplex.writeSolution`.
  - In the .NET API, see the methods `Cplex.ReadSolution` and `Cplex.WriteSolution`.
- In the Callable Library, use the new routine `CPXreadcopysol` to read SOL files. Use the changed routine `CPXsolwrite` to write SOL files.

> ***Note:*** *There is already a different XML file format available for serializing Concert Technology models and solutions. Existing classes and methods of Concert Technology already serialize modeling and solution objects; those objects are thus adapted for reading and writing in XML format. See the topic XML File Format on page 40 in the ILOG CPLEX File Formats Reference Manual and the reference manuals of the APIs for details about serialization and XML.*

NET and MIN File Formats

Improved methods and routines make it possible to read network files, that is, files containing a network structure in a model in NET format with the extension `.net` or MIN format with the extension `.min`.

- In Concert Technology, use these methods:
  - `IloCplex::importModel` in the C++ API
  - `IloCplex.importModel` in the Java API
  - `Cplex.ImportModel` in the .NET API
- In the Callable Library, use the routine `CPXreadcopyprob`.
- In the Interactive Optimizer, use the `read` command.

NET or MIN format files in Concert Technology will still generate an LP representation of a network problem.

Methods and Routines to Write LP Files Retain Names in Model

Methods of all APIs in Concert Technology and C routines in the Callable Library will modify the names of modeling elements when writing LP files (that is, formatted files with the extension `.lp`) so that the names of modeling elements remain readable. This improvement is available:

- In Concert Technology when you use the method `exportModel` with an argument that includes a filename with the `.lp` extension
- In the Callable Library when you use the routine `CPXwriteprob` either with `lp` as the file type or with `.lp` as the extension of the file name.

Modeling elements for which this improvement makes the name legible in LP format files include the following:

- Variables
- Constraints
- Ranges
- Special ordered sets (SOSs)
- Lazy constraints
- User-defined cuts
- Indicators

New Error Code

`CPXERR_XMLPARSE 1425`

XML parsing error at line %d. The parser was unable to parse the input file. Additional information about the reason is given in the message.

`CPXERR_NO_CONFLICT 1719`

No conflict is available. Either a conflict has not been computed or the computation failed.

`CPXERR_CONFLICT_UNSTABLE 1720`

Computation failed because a previously detected infeasibility could not be reproduced. A conflict exists and can be queried, but it is not minimal.

New Features in Concert Technology

- *Parameter Sets in Concert* on page 40

Parameter Sets in Concert

You can now group parameters into sets, and ILOG CPLEX will treat them together in Concert Technology, allowing you to store and restore parameters that are not at their default value in your application. For more detail, see the documentation of these classes in the reference manual:

- `IloCplex::ParameterSet` in C++
- `IloCplex.ParameterSet` in Java
- `Cplex.ParameterSet` in .NET

Semi-Continuous Variables in Concert Technology

Semi-continuous variables in Concert Technology no longer require a finite upper bound at solution time.

Improved Error Messages in Concert Technology

Error messages from Concert Technology now include the name of the class that raised the error or threw the exception, so that you can easily identify that class and consult its documentation in the reference manual for more detail about the conditions that caused the error or exception.

Serialization in Java API

Offering better support for J2EE, ILOG CPLEX now enables you to serialize modeling objects. In particular, the modeling classes of the Java API, such as `IloCplexModeler`, implement the interface `java.io.Serializable`.

Logical Constraints in Java API

Logical constraints, formerly available only through the C++ API of Concert Technology, are now available through the Java API of Concert Technology. In the Java interface

`IloModeler` (implemented by the classes `IloCplexModeler` and `IloCplex`), the following methods implement logical constraints:

- `and`
- `or`
- `not`
- `ifThen`

Logical Constraints in .NET API

Logical constraints, formerly available only through the C++ API of Concert Technology, are now available through the .NET API of Concert Technology. In the .NET interface `Modeler` (implemented by the classes `CplexModeler` and `Cplex`), the following methods implement logical constraints:

- `And`
- `Or`
- `Not`
- `IfThen`

New Method getObjective in C++ API

A new method `IloCplex::getObjective` in the C++ API returns an instance of the class `IloObjective` extracted by the invoking algorithm. In other words, this method returns the objective as an *object* (not the objective value returned by the methods `getObjValue`).

New Features in the Callable Library:

New Routines to Query Infeasible Solutions

The following new routines are available in the Callable Library to query the infeasibility of a solution:

- `CPXgetcolinfeasibility` computes the infeasibility of a given solution for a range of variables.
- `CPXgetrowinfeasibility` computes the infeasibility of a given solution for a range of linear constraints.
- `CPXgetqconstrinfeasibility` computes the infeasibility of a given solution for a range of quadratic constraints.
- `CPXgetindconstrinfeasibility` computes the infeasibility of a given solution for a range of indicator constraints.
- `CPXgetsosinfeasibility` computes the infeasibility of a given solution for a range of special ordered sets (SOSs).

See the reference manual of the Callable Library for more detail about these new query routines.

New Routine for Solution Files: CPXreadcopysol

The new Callable Library routine `CPXreadcopysol` reads an LP or QP solution from a file with the file extension `.sol` (that is, a SOL file) in XML syntax and copies that solution into a CPLEX problem object. This new routine also reads SOL files containing a MIP start.

If the advanced indicator parameter `CPX_PARAM_ADVIND` is set to 1 (one, its default) or 2, the solution read and copied by this routine will be used for an advanced start of a subsequent optimization.

The SOL file may contain these elements:

- advanced MIP start,
- basis status,
- primal values,
- dual values,
- combinations of those elements.

New Routines for Special Ordered Sets in Callable Library

New routines `CPXgetsosname` and `CPXgetsosindex` have been added to the Callable Library. `CPXgetsosname` accesses a range of names of special ordered sets in a CPLEX problem object. `CPXgetsosindex` searches for the index of a special ordered set in a CPLEX problem object.

MIP Starts in Callable Library

The routine `CPXchgmipstart` is now available in the Callable Library to modify an existing problem object for a MIP start or to create a MIP start if one does not exist already.

New Quality Values in Callable Library

There are new quality values associated with indicator constraints.

- `CPX_MAX_INDSLACK_INFEAS`
  - Numeric meaning accessed by `CPXgetdblquality`: Maximum infeasibility of the indicator constraints, or equivalently, the maximum bound violation of the indicator constraint slacks.
  - Integer meaning accessed by `CPXgetintquality`: Lowest index of the indicator constraints where the maximum indicator slack infeasibility occurs.
- `CPX_SUM_INDSLACK_INFEAS`
  - Numeric meaning accessed by `CPXgetdblquality`: Sum of the infeasibilities of the indicator constraints.
  - Integer value not meaningful.

Concert Technology does not distinguish indicator constraints from linear constraints in this respect.

New Features in the Interactive Optimizer

FeasOpt Available in Interactive Optimizer

FeasOpt (formerly available only through Concert Technology or the Callable Library) is now accessible in the Interactive Optimizer. This feature computes a minimal relaxation of the bounds of variables or the righthand sides of constraints that make the active model feasible. It is particularly useful for analyzing models that ILOG CPLEX has proved infeasible.

To invoke this feature from the Interactive Optimizer, use one of the following commands:

- `feasopt constraints`
- `feasopt variables`
- `feasopt all`

Optionally, you can instruct ILOG CPLEX to find an optimal solution of the minimally relaxed model. To do so, set the FeasOpt Mode to one of the values from Table 8 on page 44. For example, you could write:

```
set feasopt 3
```

Table 8 *FeasOpt Modes in Interactive Optimizer*

Value	Meaning
0	find relaxation with minimal sum of infeasibilities (default)
1	find optimal solution with minimal relaxation from 0
2	find relaxation with minimal number of infeasibilities
3	find optimal solution with minimal relaxation from 2
4	find relaxation with minimal sum of squared infeasibilities
5	find optimal solution with minimal relaxation from 4

There is also a parameter available through the Interactive Optimizer to control the tolerance for this feature: `set feasopt tolerance`.

There are limitations of `feasopt` in the Interactive Optimizer. The Interactive Optimizer does not attempt to relax bounds on binary (0-1) variables nor in special ordered sets (SOS).

Infeasibility as Conflict in the Interactive Optimizer

Most of the features of the conflict refiner available in the Callable Library or Concert Technology are also available in the Interactive Optimizer, with the exception of groups of constraints and preferences among groups or constraints.

To invoke the conflict refiner on an infeasible model in the Interactive Optimizer, use the command:

```
conflict
```

To display the results of the conflict refiner, use the command:

```
display conflict all
```

To write the results to a file of type CLP, use the following command, substituting the name of your file for file:

```
write file.clp
```

For more detail about the conflict refiner generally, see *Conflict Refiner: Analyzing Constraints and Bounds in Infeasible Models* on page 30 in these release notes and *Diagnosing Infeasibility by Refining Conflicts* on page 347 in the *ILOG CPLEX User's Manual*.

Longer Variable Names in Interactive Optimizer

The Interactive Optimizer is now capable of displaying longer (up to 256 characters) names of variables automatically. In other words, variable names are no longer truncated to 16 characters.

Greater Precision in MPS, REW Files in Interactive Optimizer

As noted in *Precision of Output in MPS and REW Files: MPSLongNum* on page 37, ILOG CPLEX now offers you more control over the precision of output in MPS and REW files when you read or write them in the Interactive Optimizer.

Wildcards in Interactive Optimizer

The Interactive Optimizer now recognizes wildcards:

- question mark (?) for a single character;
- asterisk (*) for zero or more characters.

Wildcards are recognized in the Interactive Optimizer in these situations:

- When you query solution information
- In `display` commands that accept a range of constraints or variables; for example, `display problem histogram` does not accept wildcards, but `display problem constraints` accepts wildcards.
- In `change` commands that accept a range of constraints or variables

Table 9 *New Parameters*

Parameter in Concert Technology	Parameter in Callable Library	Parameter in Interactive Optimizer	Purpose
ConflictDisplay	CPX_PARAM_CONFLICTDISPLAY	display conflict	Determines how much information to log and display for the conflict refiner
EpRelax	CPX_PARAM_EPRELAX	feasopt tolerance (newly available in Interactive Optimizer)	Sets epsilon used to measure relaxation in FeasOpt
EpLin	(not applicable)	(not applicable)	Sets epsilon used in linearization
FeasOptMode	CPX_PARAM_FEASOPTMODE	feasopt mode	Sets mode for FeasOpt phases and solutions
MemoryEmphasis	CPX_PARAM_MEMORYEMPHASIS	emphasis memory	Emphasizes reduced use of memory
MPSLongNum	CPX_PARAM_MPSLONGNUM	output mpslong	Controls the precision of output in MPS and REW files
NumericalEmphasis	CPX_PARAM_NUMERICALEMPHASIS	emphasis numerical	Emphasizes extreme numerical caution
PolishTime	CPX_PARAM_POLISHTIME	mip limits polishtime	Limits time in seconds spent on solution polishing after normal MIP optimization
ProbeTime	CPX_PARAM_PROBETIME	mip limits probetime	Limits time in seconds spent probing
RepairTries	CPX_PARAM_REPAIRTRIES	mip limits repairtries	Limits number of times ILOG CPLEX tries to repair an infeasible MIP start supplied by user
RepeatPresolve	CPX_PARAM_REPEATPRESOLVE	preprocessing repeatpresolve	Reapplies MIP presolve after the root is processed

Index

E

F

G

I

J

K

L