IBM ILOG CP Optimizer V2.3

Release Notes

Table of contents

Copyright notice

Trademarks

IBM, the IBM logo, ibm.com, Websphere, ILOG, the ILOG design, and CPLEX are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at *http://www.ibm.com/legal/copytrade.shtml*

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

IBM ILOG CP Optimizer V2.3 Release Notes

These release notes highlight the features in the release of IBM® ILOG® CP Optimizer V2.3. Please review these notes before using CP Optimizer V2.3.

In this section

Notes for users
Describes the few limitations of this version of CP Optimizer based on the chosen API.

Changes since CP Optimizer 2.2
Describes the changes for the release of V2.3.

Notes for users

Describes the few limitations of this version of CP Optimizer based on the chosen API.

In this section

Limitations of the Java API
Describes the limitations in the Java API.

Limitations of the Microsoft .NET Framework languages API
Describes the limitations in the .NET languages API.

Notes for Solaris users
Describes troubleshooting for Solaris users.

Limitations of the Java API

When using the Java™ API, models that make use of the class `IloCustomConstraint` cannot be solved with the parameter `IloCP.IntParam.Workers` set to a value strictly greater than 1.

Limitations of the Microsoft .NET Framework languages API

On Windows® operating systems (32 and 64 bit), the Microsoft® .NET Framework languages API is available; however, advanced features that require subclassing (custom propagators, custom selectors, custom evaluators) are not available.

Notes for Solaris users

If you have trouble launching 64-bit executables (port ultrasparc64_9_9), please make sure you have the 64-bit compiler runtime libraries in your path. You can do this by setting the environment variable `LD_LIBRARY_PATH`. For example, in `csh`:

```
setenv LD_LIBRARY_PATH /usr/lib/sparcv9:$LD_LIBRARY_PATH
```

Changes since CP Optimizer 2.2

Describes the changes for the release of V2.3.

In this section

Using IBM ILOG CP and CP Optimizer
Describes how to create a C++ application which uses both products in the same executable.

Using IBM ILOG CP and CP Optimizer

Until this version, CP Optimizer shared many class and function names with IBM® ILOG® CP: mostly those with an `Ilc` prefix and solely within the Solver component. This name sharing was done to ease porting of an IBM ILOG CP application to IBM ILOG CP Optimizer.

With this release of IBM ILOG CP Optimizer and IBM ILOG CP, you can now create a C++ application which uses both products in the same executable, allowing you to evolve your IBM ILOG CP application by adding new modules using IBM ILOG CP Optimizer.

We have renamed symbols in IBM ILOG CP Optimizer to avoid any clashes with those in IBM ILOG CP. Renaming from `IlcXXX` to `IlcCPOXXX` has been performed on internal and documented symbol names (for example `IlcIntVar` has become `IlcCPOIntVar` in IBM ILOG CP Optimizer). In order to maintain compatibility in the vast majority of cases, IBM ILOG CP Optimizer contains `#define` statements which will transform, for example, `IlcIntVar` into `IlcCPOIntVar` in your programs. We recommend that you continue to use `IlcIntVar` inside CP Optimizer for new developments. A full list of defines which affect documented CP Optimizer classes or functions is given at the end of this text.

One symbol which cannot be `#defined` is `ilc_fail_stop_here`, as it is used only in the debugger and not at compile time. This symbol has been renamed `ilccpo_fail_stop_here`.

Finally, it should be noted that although IBM ILOG CP Optimizer and IBM ILOG CP can be linked together in the same executable, code segments using the two different products cannot coexist in the same source file. Thus, in your application, each source file must use at most one of IBM ILOG CP and IBM ILOG CP Optimizer. In particular, you should never include, either directly or indirectly, header files from both products in the same source file.

List of defines in cp.h of IBM ILOG CP Optimizer V2.3

```
#define IlcAbs IlcCPOAbs
#define IlcAllocationStack IlcCPOAllocationStack
#define IlcConstraintArray IlcCPOConstraintArray
#define IlcConstraintI IlcCPOConstraintI
#define IlcConstraint IlcCPOConstraint
#define IlcDemonI IlcCPODemonI
#define IlcDemon IlcCPODemon
#define IlcExponent IlcCPOExponent
#define IlcFloatArray IlcCPOFloatArray
#define IlcFloatExpI IlcCPOFloatExpI
#define IlcFloatExp IlcCPOFloatExp
#define IlcFloatMax IlcCPOFloatMax
#define IlcFloatMin IlcCPOFloatMin
#define IlcFloatVarArray IlcCPOFloatVarArray
#define IlcFloatVar IlcCPOFloatVar
#define IlcGoalArray IlcCPOGoalArray
#define IlcGoalI IlcCPOGoalI
#define IlcGoal IlcCPOGoal
#define IlcIntArray IlcCPOIntArray
#define IlcIntExpI IlcCPOIntExpI
#define IlcIntExp IlcCPOIntExp
#define IlcIntExpIterator IlcCPOIntExpIterator
#define IlcIntPredicateI IlcCPOIntPredicateI
#define IlcIntPredicate IlcCPOIntPredicate
#define IlcIntSelectEvalI IlcCPOIntSelectEvalI
```

```
#define IlcIntSelectI IlcCPOIntSelectI
#define IlcIntSelect IlcCPOIntSelect
#define IlcIntSetArray IlcCPOIntSetArray
#define IlcIntSetI IlcCPOIntSetI
#define IlcIntSet IlcCPOIntSet
#define IlcIntSetIterator IlcCPOIntSetIterator
#define IlcIntTupleSet IlcCPOIntTupleSet
#define IlcIntVarArray IlcCPOIntVarArray
#define IlcIntVarDeltaIterator IlcCPOIntVarDeltaIterator
#define IlcIntVarI IlcCPOIntVarI
#define IlcIntVar IlcCPOIntVar
#define IlcLog IlcCPOLog
#define IlcPower IlcCPOPower
#define IlcRevAny IlcCPORevAny
#define IlcRevBool IlcCPORevBool
#define IlcRevFloat IlcCPORevFloat
#define IlcRevInt IlcCPORevInt
#define IlcSearchLimitI IlcCPOSearchLimitI
#define IloCPConstraintI IloCPOCPConstraintI
#define IloGoalFail IloCPOGoalFail
#define IloGoalI IloCPOGoalI
#define IloGoal IloCPOGoal
#define IloGoalTrue IloCPOGoalTrue
```

ILOG CP Optimizer 2.2 Release Notes

These release notes highlight the features in the release of ILOG CP Optimizer 2.2. Please review these notes before using ILOG CP Optimizer 2.2.

In this section

Notes for users
Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

Changes since CP Optimizer 2.1.1
Describes the changes for the release of 2.2.

Port changes
Summarizes new ports as well as deprecated or removed ports and recommends replacements.

Notes for users

Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

In this section

Limitations of the Java API
Describes the limitations in the Java API.

Limitations of the .NET languages API
Describes the limitations in the .NET languages API.

Notes for Solaris users
Describes troubleshooting for Solaris users.

Limitations of the Java API

Models that make use of the class `IloCustomConstraint` cannot be solved with the parameter `IloCP.IntParam.Workers` set to a value strictly greater than 1.

Limitations of the .NET languages API

On Windows 32 and 64, the .NET languages API is available; however, advanced features that require subclassing (custom propagators, custom selectors, custom evaluators) are not available.

Notes for Solaris users

If you have trouble launching 64-bit executables (port ultrasparc64_9_9), please make sure you have the 64-bit compiler runtime libraries in your path. You can do this by setting the environment variable `LD_LIBRARY_PATH`. For example, in `csh`:

```
setenv LD_LIBRARY_PATH /usr/lib/sparcv9:$LD_LIBRARY_PATH
```

Changes since CP Optimizer 2.1.1

Describes the changes for the release of 2.2.

In this section

Initial starting solution
Describes the initial starting solution.

Generalized alternative constraint
Describes the generalization of the alternative constraint.

Generalized bounds on cumulative functions
Describes the generalization of bounds on cumulative functions.

Deprecated Methods
Lists the methods that have been deprecated.

Initial starting solution

It is possible to supply an initial starting solution to CP Optimizer which can result in CP Optimizer producing better solutions earlier in the search than would otherwise be possible. This solution can be complete and valid, (which results in CP Optimizer producing this solution, and then going on to improve it), or can be incomplete and/or invalid (which results in CP Optimizer using it as a hint at how to produce a solution). For more information, see the "Starting Point" concept in the reference manual. This feature is not yet available in ILOG OPL Development Studio scripting.

Provided in the distribution are new examples that illustrate this new “starting point" API:

- In the **C++ API**, the new examples are `sched_goalprog.cpp` and `plantLocation.cpp`.
- In the **Java API**, the new examples are `SchedGoalProg.java` and `PlantLocation.java`.
- In the **C# API**, the new examples are `SchedGoalProg.cs` and `PlantLocation.cs`.

Generalized alternative constraint

The "alternative" constraint has been generalized to synchronize more than one alternative interval variable with a master interval variable. For more information, see `IloAlternative` (C++), `ilog.cp.IloCP.alternative` (Java), and `ILOG.CP.CP.Alternative` (C#). This feature is available in ILOG OPL Development Studio.

Generalized bounds on cumulative functions

In previous versions of CP Optimizer, bounds on cumulative functions were limited to being simple integers. Now they can be any constrained integral expression or variable, allowing resources capacity (or minumum level) be an unknown of the problem. For more information, see `operator<=`, `operator>=` (C++), `ilog.cp.IloCP.le` and `ilog.cp.IloCP.ge` (Java), and `ILOG.CP.CP.Le` and `ILOG.CP.CP.Ge` (C#). This feature is available in ILOG OPL Development Studio.

Deprecated Methods

The following methods have been deprecated and will be removed for the next release:

- `IlcIntVar::getMinDelta()`
- `IlcIntVar::getMaxDelta()`
- `IlcFloatVar::getMinDelta()`
- `IlcIntVar::getMaxDelta()`

Port changes

Summarizes new ports as well as deprecated or removed ports and recommends replacements.

In this section

Overview
Provides the overview to the port changes for this release.

New ports
Lists the new ports.

Deprecated ports
Lists the deprecated and removed ports.

Overview

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) see the file `yourCPHome/mptable.html` in the distribution of the product. That file contains more detailed descriptions of available ports than do these highlights. Please refer to the ILOG customer support web site for the most recent information about ports.

New ports

A port of ILOG CP Optimizer on Mac OS X is now available for Mac OS X 64–bit version 10.4 or higher . For this release, support is provided only for the C++ API. There is no support for Java on Mac OS for this release.

Deprecated ports

IBM Power AIX 5

The ports named power32_aix5.2_7.0 and power64_aix5.2_7.0 are removed. The recommended replacement is to migrate to AIX 5.3 or higher version, which is also supported by this port.

ILOG CP Optimizer 2.0 Port support changes

Port name	Platform	Status	Recommended replacement
`power32_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 32–bit	**removed: no longer available now**	AIX 5.3 or higher
`power64_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 64–bit	**removed: no longer available now**	AIX 5.3 or higher

ILOG CP Optimizer 2.1 Release Notes

These release notes highlight the features in the release of ILOG CP Optimizer 2.1. Please review these notes before using ILOG CP Optimizer 2.1.

In this section

Notes for users
Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

Changes since CP Optimizer 2.0.1
Describes the changes for the release of 2.1.

Port changes
Summarizes new ports as well as deprecated or removed ports and recommends replacements.

Notes for users

Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

In this section

Limitations of the Java API
Describes the limitations in the Java API.

Limitations of the .NET languages API
Describes the limitations in the .NET languages API.

Notes for Solaris users
Describes troubleshooting for Solaris users.

Limitations of the Java API

Models that make use of the class `IloCustomConstraint` cannot be solved with the parameter `IloCP.IntParam.Workers` set to a value strictly greater than 1.

Limitations of the .NET languages API

On Windows 32 and 64, the .NET languages API is available; however, advanced features that require subclassing (custom propagators, custom selectors, custom evaluators) are not available.

Notes for Solaris users

If you have trouble launching 64-bit executables (port ultrasparc64_9_9), please make sure you have the 64-bit compiler runtime libraries in your path. You can do this by setting the environment variable `LD_LIBRARY_PATH`. For example, in `csh`:

```
setenv LD_LIBRARY_PATH /usr/lib/sparcv9:$LD_LIBRARY_PATH
```

Changes since CP Optimizer 2.0.1

Describes the changes for the release of 2.1.

In this section

Improvements to performance
Describes some of the performance improvements.

Availability of 64–bit port for the .NET languages API
Describes the additional port.

Addition of the temporal relaxation parameter
Describes the new temporal relaxation parameter.

Resolved issues
Lists the resolved issues.

Improvements to performance

Performance improvements that have been made for this release include:

- Speed up of the propagation of the no overlap constraint
  - when there is a transition distance and a complicated objective or
  - when the parameter `IloCP::NoOverlapInferenceLevel` is set to `IloCP::Medium` or `IloCP::Extended`.
- Improved propagation of cumul functions when the parameter `IloCP::CumulFunctionInferenceLevel` is set to `IloCP::Extended`.
- Improved propagation of state functions.

Availability of 64–bit port for the .NET languages API

The .NET languages APIs are now available on Windows 64.

Addition of the temporal relaxation parameter

A new advanced parameter, `IloCP::TemporalRelaxation`, can be used to control the usage of a temporal relaxation internal to the engine. This parameter can take values `IloCP::On` or `IloCP::Off`, with `IloCP::On` being the default, meaning the relaxation is used in the engine when needed. For some models, using the relaxation becomes inefficient, and you may deactivate the use of the temporal relaxation using value `IloCP::Off`.

Resolved issues

These are the registered issues that are fixed in ILOG CP Optimizer 2.1 since the previous release (CP Optimizer 2.0.1).

ILOG CP Optimizer Resolved Issues

Issue ID	Issue
CP-173	Missing API in Java and C# to access cumul function value
CP-174	Missing API to access state function solution values
CP-177	Occasional freeze after the second solution in a scheduling model
CP-183	numberOfSegments gives different result (OPL/CP)
CP-198	getNumberOfSegments crashes if the cumulFunction is atomic
CP-200	Crash during OPL profiling
CP-201	Missing propagation in cumulative timetable
CP-202	Missed solutions with DFS search
CP-204	Missing aggregation of logical constraints in the case of complex forall
CP-208	CP Optimizer claims a solution is optimal whereas it isn't
CP-213	Missing propagation of state function with transition distance and thus missing solutions
CP-216	Time limit seem not to be taken into account for initial propagation
CP-218	Possible crash when re-extracting a transition distance previously extracted in CP Optimizer
CP-220	Internal error when exporting the value of a goal after solving a scheduling model
CP-221	JVM crash with a scheduling model

These are the registered issues for CP Optimizer when used from OPL Development Studio that are fixed in ILOG CP Optimizer 2.1 since the previous release (CP Optimizer 2.0.1)

ILOG CP Optimizer from OPL Development Studio Resolved Issues

Issue ID	Issue
OPL-3543	Aborting a CP solve exhausts memory
OPL-3552	CP param Workers isn't properly set using .ops
OPL-3750	Problem with evaluation of pwFct[1](2)
OPL-3869	Wrong extraction of transition distance for state functions

Port changes

Summarizes new ports as well as deprecated or removed ports and recommends replacements.

In this section

Overview
Provides the overview to the port changes for this release.

Deprecated ports
Lists the deprecated and removed ports.

Overview

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) see the file `yourCPHome/mptable.html` in the distribution of the product. That file contains more detailed descriptions of available ports than do these highlights. Please refer to the ILOG customer support web site for the most recent information about ports.

Deprecated ports

IBM Power AIX 5

The ports named power32_aix5.2_7.0 and power64_aix5.2_7.0 are deprecated. IBM announces plans to stop support for AIX 5.2 in the next twelve months. The recommended replacement is to migrate to AIX 5.3 or higher version, which is also supported by this port.

ILOG CP Optimizer 2.0 Port support changes

Port name	Platform	Status	Recommended replacement
`power32_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 32–bit	**deprecated**: no longer available in the next release	AIX 5.3 or higher
`power64_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 64–bit	**deprecated**: no longer available in the next release	AIX 5.3 or higher

ILOG CP Optimizer 2.0.1 Release Notes

In this section

Known issues corrected in technical release 2.0.1

- An exception is now thrown if you define an interval variable with a minimum start lower than `IloIntervalMin (-IloIntMax div 2 + 1)` or a maximum end higher than `IloIntervalMax (IloIntMax div 2 - 1)`.
- A new parameter, `TemporalRelaxationLevel`, can be used to control the strength of usage of a temporal relaxation internal to the engine. This parameter can take values in the set 0, 1, 2 with 2 being the default, meaning the strongest usage of the relaxation. For some models, using the strongest enforcement becomes inefficient, and you may deactivate the use of the temporal relaxation entirely using:

  in C++ `cp.setParameter(IloCP::TemporalRelaxationLevel, 0)`

  in Java `cp.setParameter(IloCP.IntParam.TemporalRelaxationLevel, 0)`

  in C# `cp.SetParameter(CP.IntParam.TemporalRelaxationLevel, 0)`

- On 64-bit architectures, some problems were encountered with stepwise functions (`IloNumToNumStepFunction`). These problems have been resolved.
- Memory consumption of multiple constraints of type `IloForbidStart`, `IloForbidEnd` and `IloForbidExtent`, which share the same stepwise function (`IloNumToNumStepFunction`), has been reduced.
- The `IloNoOverlap` constraint experienced some problems when setup times were specified and the sequence variable that was passed contained no present intervals in the solution. These problems have been resolved.
- Models where a cumul function expression (`IloCumulFunctionExpr`) appeared as a term in more than one other cumul function expression were not extracted correctly. This has been corrected.
- Some of the data files for flexible job-shop instances (in `<CPO_dir>\examples\data\jobshopflex*.data`) were incorrect. The data is now correct.
- A memory leak that occurred when you deleted a sequence variable (`IloIntervalSequenceVar::end()`) in a Concert technology environment has been fixed.
- In some circumstances, a solution would be missed when sequence variables involved zero-length intervals. This problem has been fixed.

ILOG CP Optimizer 2.0 Release Notes

These release notes highlight the features in the release of ILOG CP Optimizer 2.0. Please review these notes before using ILOG CP Optimizer 2.0.

In this section

What is new in ILOG CP Optimizer 2.0?
Describes the new features in CP Optimizer 2.0.

Notes for users
Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

Changes since CP Optimizer 1.1
Describes the changes for the release of 2.0, other than the introduction of scheduling.

Obsolete, deprecated, and updated ports
Summarizes new ports as well as deprecated or removed ports and recommends replacements.

What is new in ILOG CP Optimizer 2.0?

ILOG CP Optimizer 2.0 is a major milestone – the introduction of scheduling. ILOG CP Optimizer enables you to use a 'model and run' development process for solving and optimizing scheduling problems with fine-grained time. For this, ILOG CP Optimizer introduces a new modeling framework, based on concepts such as intervals and functions, that harmonizes the expression of typical scheduling constraints such as precedence, cumul, state, sequence with transition times and calendar. ILOG CP Optimizer also includes a robust search algorithm to solve and optimize scheduling problems. To learn about this new feature, please refer to the The *ILOG CP Optimizer Reference Manuals* and the *Getting Started with ILOG CP Optimizer Manual*.

Notes for users

Describes the few limitations of this version of ILOG CP Optimizer based on the chosen API.

In this section

Limitations of the Java API
Describes the limitations in the Java API.

Limitations of the .NET languages API
Describes the limitations in the .NET languages API.

Notes for Solaris users
Describes troubleshooting for Solaris users.

Use of CPLEX
Describes the manner in which CP Optimizer uses CPLEX.

Limitations of the Java API

For this release, setting the `IloCP.IntParam.Workers` parameter to value other than 1 in the Java API is not currently supported.

Limitations of the .NET languages API

The .NET languages API is not available on Windows 64.

On Windows 32, the .NET languages API is available; however, advanced features that require subclassing (custom propagators, custom selectors, custom evaluators) are not available.

Notes for Solaris users

If you have trouble launching 64-bit executables (port ultrasparc64_9_9), please make sure you have the 64-bit compiler runtime libraries in your path. You can do this by setting the environment variable `LD_LIBRARY_PATH`. For example, in `csh`:

```
setenv LD_LIBRARY_PATH /usr/lib/sparcv9:$LD_LIBRARY_PATH
```

Use of CPLEX

This version of CP Optimizer uses ILOG CPLEX to aid in the resolution of certain scheduling problems. For this, you must link your application with the ILOG CPLEX libraries provided in this installation of CP Optimizer.

Other than linking with CPLEX, the use of CPLEX by CP Optimizer is completely automatic and invisible. The commands for linking CP Optimizer to produce executables are provided in the example project files or makefiles.

Changes since CP Optimizer 1.1

Describes the changes for the release of 2.0, other than the introduction of scheduling.

In this section

Addition of the time mode parameter
Describes the time mode parameter.

Resolved issue
Lists the resolved issue.

Addition of the time mode parameter

CP Optimizer uses time for both display purposes and for limiting the search. These timings can be measured either by CPU Time or by elapsed time, and the time mode parameter defines how time is measured in CP Optimizer.

When multiple processors are available and the number of workers is greater than one, then the CPU time can be greater than the elapsed time by a factor up to the number of workers.

In the **C++ API** of CP Optimizer, the time mode is controlled with the parameter `IloCP::TimeMode`. A value of `IloCP::CPUTime` indicates that time should be measured as CPU time, `IloCP::ElapsedTime` indicates that time should be measured as elapsed time. The default is `IloCP::CPUTime`.

In the **Java API** of CP Optimizer, the time mode is controlled with the parameter `IloCP.IntParam.TimeMode`. A value of `IloCP.ParameterValues.CPUTime` indicates that time should be measured as CPU time, `IloCP.ParameterValues.ElapsedTime` indicates that time should be measured as elapsed time. The default is `IloCP.ParameterValues.CPUTime`.

Likewise, in the **C# API** of CP Optimizer, the time mode is controlled with the parameter `CP.IntParam.TimeMode`. A value of `CP.ParameterValues.CPUTime` indicates that time should be measured as CPU time, `CP.ParameterValues.ElapsedTime` indicates that time should be measured as elapsed time. The default is `CP.ParameterValues.CPUTime`.

Resolved issue

This is the registered issue that is fixed in ILOG CP Optimizer 2.0 since the previous release (CP Optimizer 1.1).

ILOG CP Optimizer Resolved Issues

Issue ID	Issue
CP-146	Limits do not work properly in parallel mode

Obsolete, deprecated, and updated ports

Summarizes new ports as well as deprecated or removed ports and recommends replacements.

In this section

Overview
Provides the overview to the port changes for this release.

New ports
Lists the new ports.

Deprecated and removed ports
Lists the deprecated and removed ports.

Overview

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) see the file `yourCPHome/mptable.html` in the distribution of the product. That file contains more detailed descriptions of available ports than do these highlights. Please refer to the ILOG customer support web site for the most recent information about ports.

New ports

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) see the file `yourCPHome/mptable.html` in the distribution of the product. That file contains more detailed descriptions of available ports than do these highlights.

The following new ports are available with this release:

- New ports, suitable for x86 and x86-64 processors, and based on the Debian 4.0 distribution of the Linux operating system are available in this release. They are compatible with Red Hat Enterprise Linux 5 (RHEL5) and Suse Linux Enterprise Server 10 (SLES10).
- New ports supporting Sun UltraSPARC processors and Sun Studio 9 or higher compilers and development environment are available for 32.bit and for 64.bit architectures.
- A new port compatible with Microsoft Visual Studio 2008 is available.

ILOG CP Optimizer 2.0 Summary of new ports

Port name	Platform	Comments
`x86_debian4.0_4.1` `x86-64_debian4.0_4.1`	Debian 4 GNU/Linux Red Hat Enterprise Linux 5 (RHEL5) Suse Linux Enterprise Server 10 (SLES10)	
`ultrasparc32_9_9` `ultrasparc64_9_9`	Sun Solaris 9 or higher operating system Sun Studio 9 or higher compiler	
`x86_windows_vs2008` `x64_windows_vs2008`	x86 or x64 machine architecture Microsoft Visual Studio 2008 compiler NET Framework 2.0 or higher	

Deprecated and removed ports

IBM Power AIX 5

The ports named power32_aix5.2_7.0 and power64_aix5.2_7.0 are deprecated. IBM announces plans to stop support for AIX 5.2 in the next twelve months. The recommended replacement is to migrate to AIX 5.3 or higher version, which is also supported by this port.

Red Hat Enterprise Linux 4

Ports based on Red Hat Enterprise Linux version 4 (that is, those with _rhel4_ in their name) are deprecated and have been removed.

The recommended replacements, named x86_debian4.0_4.1 and x86-64_debian4. 0_4.1, are compatible with Red Hat Enterprise Linux version 5, Suse Linux Enterprise Server version 10 and Debian version 4.

Microsoft Visual Studio 2003

The ports based on Visual Studio 2003 have been **removed**. Recommended replacements are Microsoft Visual Studio 2005 and Microsoft Visual Studio 2008.

ILOG CP Optimizer 2.0 Port support changes

Port name	Platform	Status	Recommended replacement
`power32_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 32–bit	**deprecated**: no longer available in the next release	AIX 5.3 or higher
`power64_aix5.2_7.0`	IBM Power AIX 5.2 IBM XL C/C++ 7.0 64–bit	**deprecated**: no longer available in the next release	AIX 5.3 or higher
`x86_rhel4_*`	Red Hat Enterprise Linux 4	**deprecated** and **removed: no longer available now**	Debian 4, Red Hat Enterprise Linux 5, or Suse Linux Enterprise Server SLES10
`x86_.net2003*`	Microsoft Visual Studio 2003	**removed: no longer available now**	`x86_windows_vs2008` or `x64_windows_vs2008`

Index