The WIT Guide Supplement

A Supplement to the Book,

"Constrained Materials Management and Production Planning Tool:

User's Guide and Reference",

Also Known as "The WIT Guide".

August 25, 2020

Preface

About this document

This document is a supplement to the book, "Constrained Materials Management and Production Planning Tool: User's Guide and Reference", which is the reference manual for the open-source software tool called, "Constrained Materials Management and Production Planning Tool". That tool was historically called "WIT" and still is called "WIT" much of the time. Its reference manual is commonly called "The WIT Guide". So this document is a supplement to the WIT Guide,

The WIT Guide was written in FrameMaker by IBM employees. Now that IBM has made WIT open-source, the team that is working on it doesn't have a FrameMaker license (They are expensive.), so we (the WIT team) are treating the WIT Guide as a fixed document and just using the PDF version of it, wit.pdf. We actually used a PDF editor to make some small, essential revisions to wit.pdf, but beyond that, we expect to make no further changes to the WIT Guide. On the other hand, WIT itself is being revised, so we need to provide user documentation to the revised code. That is the purpose of this document: to provide user documentation for the aspects of WIT that have changed since it was made open-source.

How this document is organized

For clarity, the chapters of this document correspond directly to the chapters of The WIT Guide. So for example, Chapter 1 explains new or revised features of WIT, Chapter 2 defines new or revised attributes of WIT's data objects, etc.

Summary of WIT Software Changes

<To be written.>

Chapter 1: Introducing WIT

Additional Capabilities of WIT

(Capabilities that have been added to WIT since it was made open-source.)

Using COIN Solvers to Solve the Optimization Problem

In the closed-source version of WIT, during optimizing implosion, WIT would solve its optimization problem (either LP or MIP) by invoking CPLEX. In the open-source version, CPLEX can still be used for this purpose, but a new alternative is now available: WIT can now use solvers from COIN-OR to solve its optimization problem. COIN-OR is a community that creates open-source implementations of Operations Research algorithms. For information on COIN-OR, see:

`https://www.coin-or.org/`

Specifically, WIT uses the following solvers from COIN-OR: CLP and CBC. When WIT is using its COIN solvers:

- It invokes CLP through its OSI interface to solve WIT's LP problem.
- It invokes CBC (with CLP as the LP solver) to solve WIT's MIP problem.

If WIT is to use the COIN solvers to solve its optimization problem, it must be built with with the COIN solvers embedded. This is called building WIT with COIN embedded. Specifically, WIT can be built in the following four ways, regarding solvers:

- With CPLEX embedded and not COIN
- With COIN embedded and not CPLEX
- With both CPLEX and COIN embedded
- With neither CPLEX or COIN embedded

WIT selects the solver to use based on which solver(s) are embedded and on the value of a global boolean input attribute, preferCoin. Specifically, when optimizing implosion is invoked, WIT selects the solver to use by the following rule:

- If WIT was built with both COIN and CPLEX embedded and preferCoin is TRUE, the COIN solvers are used.
- If WIT was built with both COIN and CPLEX embedded and preferCoin is FALSE, CPLEX is used.
- If WIT was built with COIN embedded and not CPLEX, the COIN solvers are used.
- If WIT was built with CPLEX embedded and not COIN, CPLEX is used.
- If WIT was built with neither COIN nor CPLEX embedded, a severe error is issued.

There are two other attributes relevant to the selection of solver: coinEmbedded and cplexEmbedded. Both of these are global boolean output attributes:

- coinEmbedded is TRUE, if and only if COIN is embedded into the current build of WIT.
- cplexEmbedded is TRUE, if and only if CPLEX is embedded into the current build of WIT.

WIT has several data attributes that contain "cplex" in their name, e.g, cplexStatusCode. These attributes are specific to CPLEX do not apply to the COIN solvers. Other than those CPLEX-specific attributes, when the WIT Guide mentions CPLEX, the statements made should be re-interpreted to apply to whichever solver is being used, CPLEX or COIN.

When the COIN solvers are selected, they are used for any WIT capability that requires optimization:

- Optimizing Implosion with LP
- Optimizing Implosion with MIP
- Accelerated Optimizing Implosion
- Optimizing Implosion with Multiple Objectives
- Stochastic Implosion

Chapter 2: WIT Data Attributes

(Attributes that have been added to WIT since it was made open-source.)

Global (WIT Problem) Attributes

coinEmbedded

Output

Boolean

TRUE, iff the COIN solvers were embedded into the current build of WIT.

preferCoin

Input

Boolean

Default Value: FALSE

If both COIN and CPLEX are embedded into the current build of WIT, then:

- WIT will use the COIN solvers to solve optimization problems, when preferCoin is TRUE.
- WIT will use CPLEX to solve optimization problems, when preferCoin is FALSE.

If only one of the solvers (COIN or CPLEX) is embedded into the current build of WIT, or none is, then this attribute has no effect.

Chapter 3: Using the WIT Stand-Alone Executable

There are no revisions to WIT that need to be documented in this chapter.

Chapter 4: Using the API: Application Program Interface

Table 6

Which Attributes Can Be Changed While Preserving an Accelerated State

(Attributes that have been added to WIT since it was made open-source.)

Attribute (Input Attributes Only)	**Object Type**	**Can this attribute be changed while preserving an accelerated state?**
preferCoin	Global	No

Chapter 5: API Function Library

Global Data Functions

witGetAttribute

These are the `witGetAttribute` functions that have been added to WIT since it was made open-source:

```
witReturnCode witGetCoinEmbedded
(   WitRun * const theWitRun,
    witBoolean *   coinEmbedded);
witReturnCode witGetPreferCoin
(   WitRun * const theWitRun,
    witBoolean *   preferCoin);
```

witSetAttribute

These are the `witSetAttribute` functions that have been added to WIT since it was made open-source:

```
witReturnCode witSetPreferCoin
(   WitRun * const theWitRun,
    const witBoolean preferCoin);
```