

IBM Cognos TM1 TI-Process Synchronization: Proven practices, Recommendations and a plug&play TI-Synchronization Utility

Prepared:
Jan/Feb 2014

Created By:
Andreas Kugelmeier
Executive Consultant, FOPM
Planning Analytics Architect
IBM Data and AI Expert Labs
Mobile Phone: +1-215-384-7302
Email: kugelmeier@us.ibm.com

Notices & Disclaimers

Copyright © 2015 by International Business Machines Corporation (IBM). No part of this document may be reproduced or transmitted in any form without written permission from IBM.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations and papers (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. THIS document is distributed "AS IS" without any warranty, either express or implied. In no event shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity. IBM products and services are warranted according to the terms and conditions of the agreements under which they are provided.

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com, Aspera®, Bluemix, Blueworks Live, CICS, Clearcase, Cognos®, DOORS®, Emptoris®, Enterprise Document Management System™, FASP®, FileNet®, Global Business Services®, Global Technology Services®, IBM ExperienceOne™, IBM SmartCloud®, IBM Social Business®, Information on Demand, ILOG, Maximo®, MQIntegrator®, MQSeries®, Netcool®, OMEGAMON, OpenPower, PureAnalytics™, PureApplication®, pureCluster™, PureCoverage®, PureData®, PureExperience®, PureFlex®, pureQuery®, pureScale®, PureSystems®, QRadar®, Rational®, Rhapsody®, Smarter Commerce®, SoDA, SPSS, Sterling Commerce®, StoredIQ, Tealeaf®, Tivoli®, Trusteer®, Unica®, urban{code}®, Watson, WebSphere®, Worklight®, X-Force® and System z® Z/OS, are trademarks of International Business Machines Corporation, 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: www.ibm.com/legal/copytrade.shtml.

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.
- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.
- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.
- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Document Version History

Date	Version	Author	Description
2013	.9	Andreas Kugelmeier	Development of Utility
February, 2014	1.0	Andreas Kugelmeier	Document Release
08/07/2015	2.0	Andreas Kugelmeier	Misc. edits and improvements, add TOC

Contents

1. TI-Process Synchronization - Introduction _____ **4**

2. Recommendations on how to use SYNCHRONIZED() _____ **5**

3. Configurable TI Synchronization Utility _____ **6**

4. TI Synchronization Utility Objects _____ **8**

1. TI-Process Synchronization - Introduction

For TI processes that edit/update dimensions and/or update security, it is a proven practice to implement of a Synchronization Methodology to serialize TI processing to prevent thrashing/rollback: Thrashing, or successive rollbacks and retries, can result from the concurrent execution of Turbo Integrator processes that lock the same objects and block each other. If two or more TI processes may perform an update of one and the same dimension, the first process to acquire the dimension 'lock' will block the other process from continuing, the 2nd process will hence do a rollback of its actions before encountering the lock and then will attempt to start anew (possibly hitting the same lock later should process 1 not have finished). Such two processes should be kept from being executed in parallel and instead should only be allowed to be run in serial to avoid thrashing and excessive rollback actions. The same applies to fact updates and dimension updates: if you are running a process that updates fact data you should not attempt to update the dimension master data using a different process as this will cause locking and rollbacks. Typically, a TI process will only perform actual updates at or towards the end of the process after 1st retrieving data and/or performing various validations, and if it then finds that the object it is trying to update is locked, then it performs a roll back and retry. In some cases, two or more TI-process may even enter into a 'loop' of multiple to endless rollbacks and retries due to locking one another during various stages of processing.

Thrashing Examples:

1. Two or more TI processes may perform an update of one and the same dimension
2. The first process to acquire the dimension 'lock' will block the other process from continuing
3. The 2nd process will hence do a rollback of its actions before encountering the lock and then will attempt to start anew
4. The 2nd process may possibly hit the same lock later should process 1 not have finished or have released the lock (again resulting in a roll-back, i.e. back to the beginning...).
5. Or the 2nd process will find that the lock was released, but: the 2nd TI process ,may obtain a lock on a different object that is needed by the first process (but later on, i.e. towards the end of the 1st process running
6. Now the 2nd process is locking the 1st process & the 1st process will do a roll-back...
7. Not only may there be roll-backs and retries, but the rollbacks and retries – depending on the TI procedures in place, may affect one another, resulting in two or more processes locking each other out.

The idea behind the synchronization and serialization logic is therefore to reduce if not eliminate thrashing inefficiencies by 'locking' the target object (or a common related object) **upfront before** retrieving data or performing any other time consuming operations. A semaphore logic can be used to synchronize certain (applicable) TI processes to run in serial execution mode only. The synchronization logic will ensure that a process is told to 'wait' at the very beginning of execution (before it would require a roll-back or data updates etc.) until the 'lock' that would prevent the process from continuing is released.

As of TM1 10, the SYNCHRONIZED()¹ function allows for easy serialization of TI processes if needed.

¹ [TM1 SYNCHRONIZED\(\) function](#): IBM® Cognos® TM1® TurboIntegrator (TI) function called synchronized() can be used in a TurboIntegrator script to force serial execution of a designated set of TurboIntegrator processes. The synchronized() function uses the following syntax: synchronized(string)

A TurboIntegrator process may make any number of calls to synchronized(), with any number of lock objects. Serializing is effective from the time synchronized() is called, until the containing transaction completes. For example, if synchronized() is called from a subprocess (Ps) of master process (Pm) or master chore (Cm), the Lock Object is "released" when Pm or Cm completes. The exception is that a SaveDataAll (SDA) prematurely "ends" a transaction mid-process execution; this applies to Lock Objects as well. The synchronized() call may be placed anywhere within a TurboIntegrator script, but serialization applies to the entire

2. Recommendations on how to use SYNCHRONIZED()

SYNCHRONIZED() takes a single required parameter that is a user-defined name for a lock object. This lock object name can be used in multiple TI-processes in order to serialize their execution as a group. A TI-process may make any number of calls to `synchronized()`, with any number of lock objects. Serializing is effective from the time `synchronized()` is called, until the containing transaction completes.

Recommended use of `SYNCHRONIZED()`:

- Use **Dimension Names as the lock names for SYNCHRONIZED()**
- For TI processes that are built specifically to update a certain dimension, add the following code to the beginning of the TI process (TI prolog): **SYNCHRONIZED (<DimensionName>);**
- For TI-processes that update a cube, issue `SYNCHRONIZED()` calls for each of the cube's dimensions. If the TI is dynamic in terms of which cube it uses, use a `WHILE` loop with `TABDIM` to loop through each dimension and call `SYNCHRONIZED()`, like:

```
nFlag = 1;
nDimCounter = 1;
WHILE ( nFlag = 1 );
  sDimensionXName = "";
  sDimensionXName = TABDIM ( pSourceCube, DimCounter );
  IF ( sDimensionXName @<> " );
    SYNCHRONIZED ( sDimensionXName );
  ELSE;
    nFlag = 0;
  ENDIF;
  nDimCounter = nDimCounter + 1;
END;
```

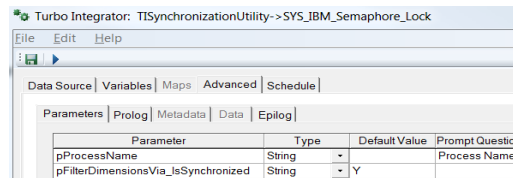
- For TI processes that are parameterized such that multiple or varying cubes, dimensions or objects could be accessed and locked, implementation of a configurable TM1 **TI-Synchronization utility** is recommended. The utility is to leverage the `SYNCHRONIZED()` function in combination with a configurable Lookup model to allow setting, maintaining and testing different serialization configurations in an ad-hoc/dynamic fashion. Such a utility – which allows a plug&play implementation – is described in the following section.

TurboIntegrator process when it is encountered. Consider a TurboIntegrator process with a `synchronized()` call somewhere in the "middle" of its script, and an operation O1 preceding that call. Two instances of this TurboIntegrator process may start at the same time. It is possible for one instance to run to completion, including its call to `synchronized()`, before the second instance reaches its `synchronized()` call. In this case, the two processes appear to the user to have run concurrently. If, instead, the second process does reach its `synchronized()` call before the first completes, it will undo any work it had done (O1) and wait for the first to complete. In this case, the two processes appear to the user to have serialized. To avoid such confusion, and to optimize the use of `synchronized()`, it is recommended (but not enforced) that `synchronized()` calls be the first statements of a TurboIntegrator process.

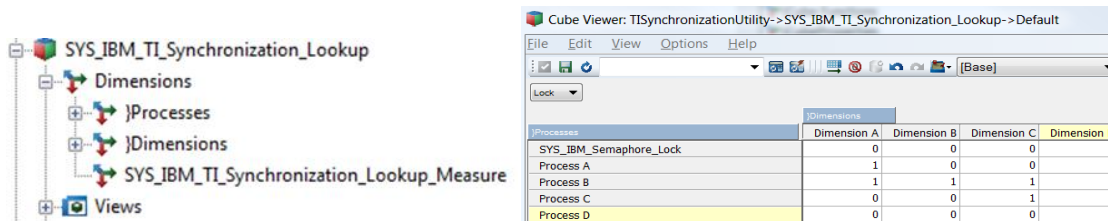
3. Configurable TI Synchronization Utility

The TM1 Utility leverages the SYNCHRONIZED() function in combination with a configurable Lookup model to allow setting, maintaining and testing different serialization configurations in an ad-hoc/dynamic fashion. Here is how to implement the TM1 Asset for semaphore locking:

- a) Drop the files from the attached archive in your TM1 data directory and restart the TM1 Database Process
- b) 'SYS_IBM_Semaphore_Lock.pro' should be invoked in the prolog of TI processes that shall be serialized with other specific or non-specific TI processes. Corresponding code to be inserted into the prolog of a TI process:
EXECUTEPROCESS ('SYS_IBM_Semaphore_Lock' , 'pProcessName' , GetprocessName());



- c) For dimension }Dimensions.dim, create an Attribute 'Is Synchronized'. Set the attribute value for Y for every dimension for which you want to implement serialization of dimension maintenance tasks.
- d) The lookup cube 'SYS_IBM_TI_Synchronization_Lookup.cub' then is used to determine the SYNCHRONIZED() flags to be set by a specific TI process. The SYNCHRONIZED flags are set by process 'SYS_IBM_Semaphore_Lock.pro'. Configure SYS_IBM_TI_Synchronization_Lookup.cub' such that for each TI process that shall be serialized, the cube value for the corresponding dimension (i.e. the dimension the TI will edit or lock) is set to 1.



- e) Sub-process 'SYS_IBM_Semaphore_Lock.pro' will now, via the lookup cube entries in 'SYS_IBM_TI_Synchronization_Lookup.cub' set SYNCHRONIZED(<DimensionName>) flags as follows:
 - i. Cube 'SYS_IBM_TI_Synchronization_Lookup.cub' will be queried for the specific process (with process = <processname> below). The process will query the 'Lock' value for applicable <processname>/<dimensions> combinations
 - ii. Depending on process parameter pFilterDimensionsVia_IsSynchronized:
 - If pFilterDimensionsVia_IsSynchronized = N: with <dimensions> = all dimensions in }Dimensions.dim (i.e. If pFilterDimensionsVia_IsSynchronized = N or <> Y, all dimensions, i.e. all elements in }Dimensions.dim will be evaluated)
 - if pFilterDimensionsVia_IsSynchronized = Y or <dimensions> = dimensions in }Dimensions.dim with value Y for attribute 'Is Synchronized' (i.e. If Parameter

pFilterDimensionsVia_IsSynchronized = Y a filter will be applied such that a synchronization flag is only going to be evaluated against dimensions where the }Dimensions.dim Attribute 'Is Synchronized' = Y (via an MDX subset).

- ii. IF the 'Lock' value is 1, the process 'SYS_IBM_Semaphore_Lock.pro' will issue a synchronization semaphore flag <dimension> using the TM1 function SYNCHRONIZED(), i.e. SYNCHRONIZED (<dimension>) will be issued for each <processname>/<dimension> combination with value 'Lock' = 1, i.e. if needed, more than one synchronization flags can be issued by a process.

4. TI Synchronization Utility Objects

<https://ibm.box.com/s/euluj72iqi9nga1yj52fklezvd2mu99l>