

BOPF Performance Training

SAP AG, 2012



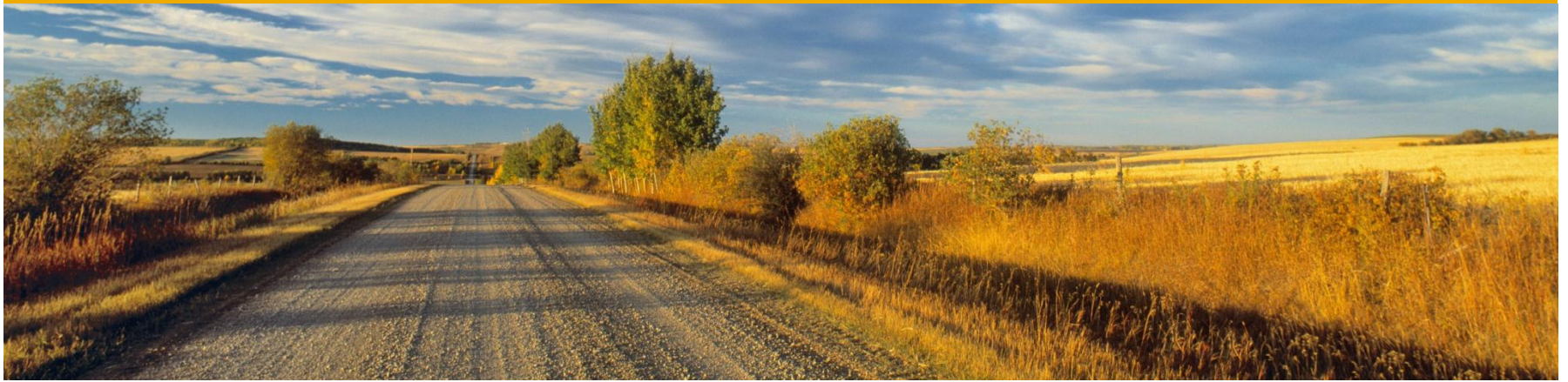
Disclaimer

This presentation outlines our general product direction and should not be relied on in making a purchase decision. This presentation is not subject to your license agreement or any other agreement with SAP. SAP has no obligation to pursue any course of business outlined in this presentation or to develop or release any functionality mentioned in this presentation. This presentation and SAP's strategy and possible future developments are subject to change and may be changed by SAP at any time for any reason without notice. This document is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP assumes no responsibility for errors or omissions in this document, except if such damages were caused by SAP intentionally or grossly negligent.

BOPF Performance

Content

1. Mass Data Processing
2. Alternative Keys
3. Determinations
4. Validations
5. Transient Data
6. Transactions
7. Properties
8. Loading Groups



Mass Data Processing

Mass Data Processing

Scenario

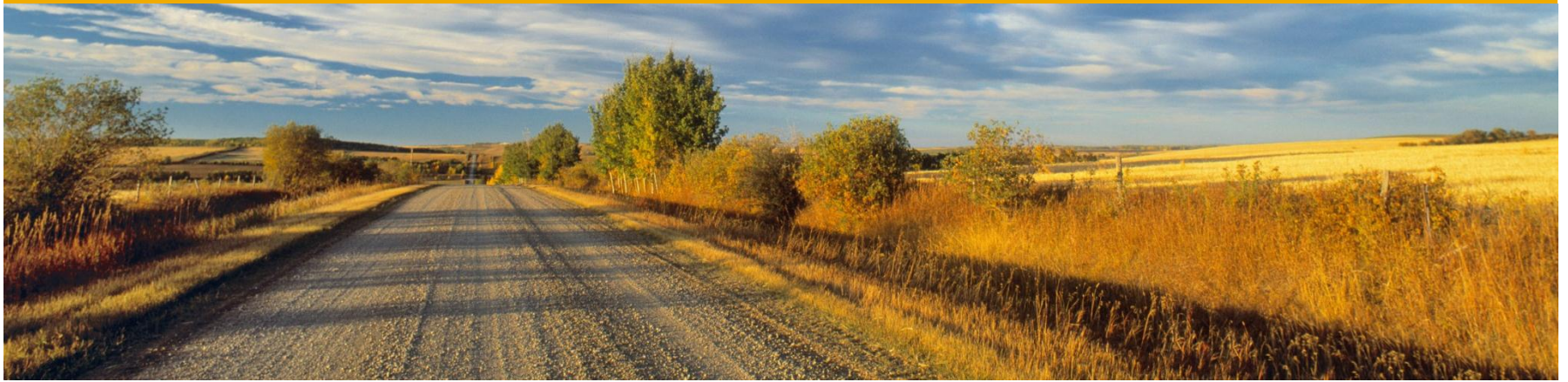
The execution of mass data scenarios (e.g. processing more than 100.000 instances by the help of a daily batch job) using the default simple buffer might lead to performance issues or out even of memory dumps.

Problem

The simple buffer has an unrestricted cache size. If the internal buffer table grows during the scenario, access to that buffer is more time consuming. If this internal table needs more memory than available, an out of memory dump occurs.

Solutions

- Implement a buffer with a maximum cache size (e.g. circular buffer principle)
- Use the simple buffer but package your workload and regularly clean your buffer after each processed package (“cleanup” core service after each “save” core service call).



Entities

Mass-Enabled Core Services

Scenario

It is important that core services, that query data from the database, are used in a mass-enabled fashion. If for instance a RETRIEVE is done in a loop, each iteration will cause a database select. If the loop iterates on many instances, this will slow down the application.

Solutions

All methods of the service manager and the io_read are mass-enabled. Fetch all data in an internal table and loop afterwards over the entries.

Database Index

Scenario

The reading access via alternative key and the navigation to child nodes are very slow and need to be accelerated.

Solutions

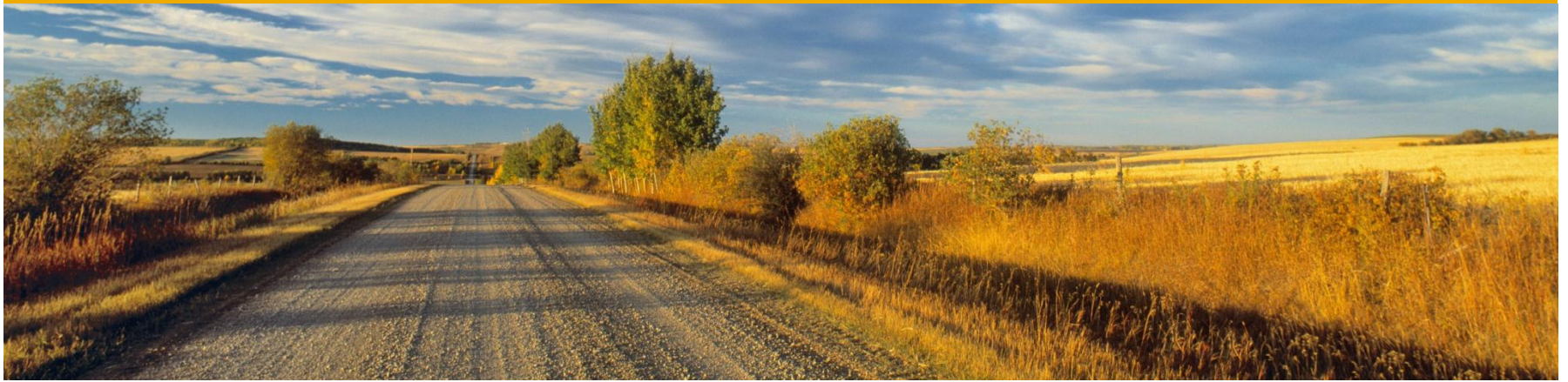
A database index improves the performance of reading data from a database table. In principle it is a (sorted) selection of database columns. If the data is changed, the corresponding index will be automatically updated which costs additional performance. There are two different kinds of an index:

A. Primary Database Index

BOPF node tables have the instance key (fields MANDT and DB_KEY) as primary key and thus automatically as primary index

B. Secondary Database Index

- Subnodes should have an index on the fields MANDT and PARENT_KEY to speed up compositions (use “Generate DDIC”>”DB-Index” checkbox)
- Alternative key access can be accelerated by a secondary database index
- Remark for Hana DB: please compare to DB guidelines, if creation of DB Index makes sense at all. E.g. when more than 500k entries are expected and when Index shall only be generated for a single column.



Alternative Keys

Secondary Database Index for Alternative Keys

Scenario

A new alternative key is created on a node. It is recommended to create a secondary index for that alternative key.

Reasons to create an Alternative Key

- Target-Resolved Binding
- The node instances are predominantly read and there is less modification

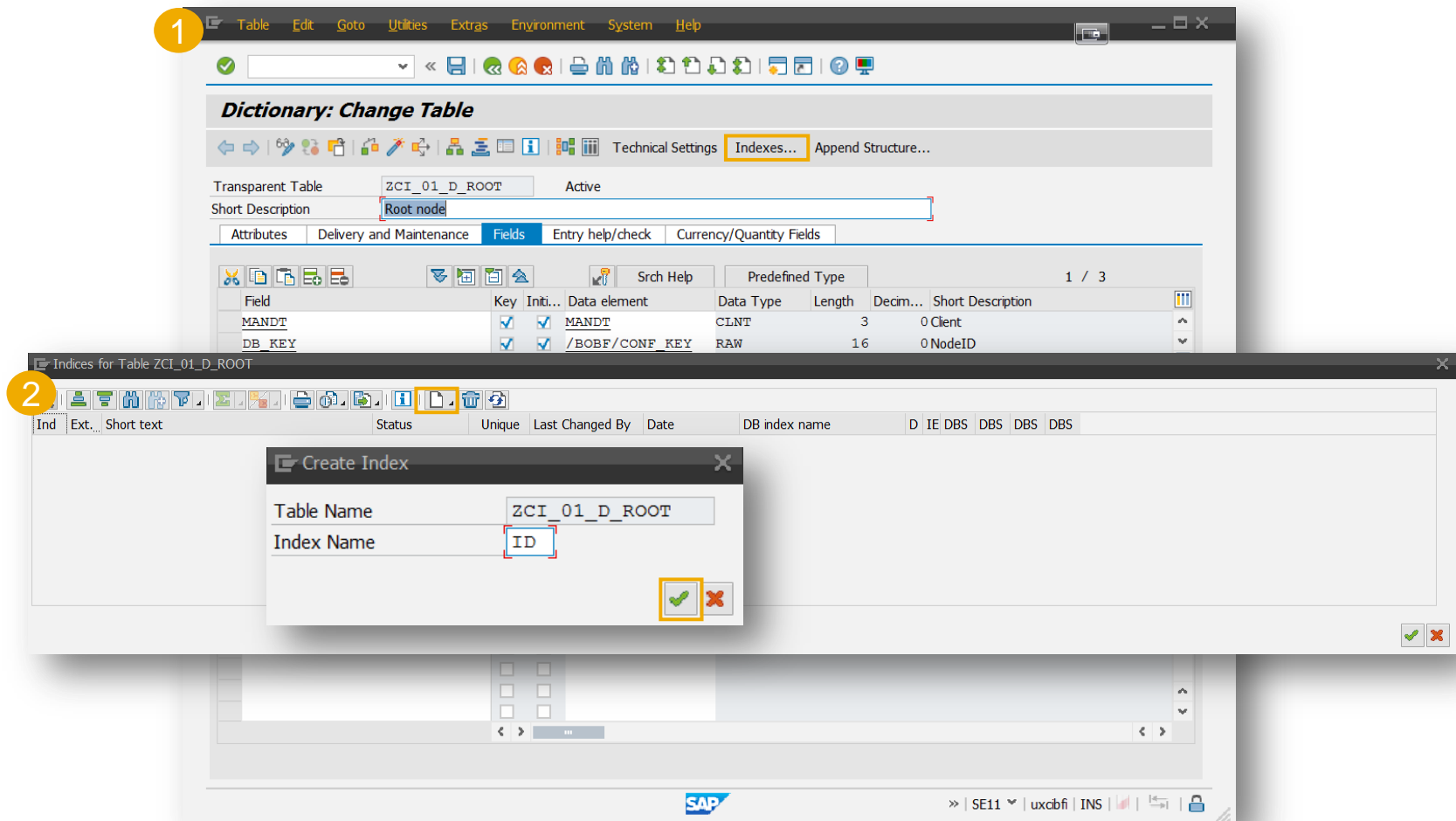
Restrictions

- Selectivity: Each alternative key value should correspond to less than 5% of the node instances
- The alternative key should consist of less than 4 fields and those should not overlapping fields of other index
- There should be only less than 5 index on the node table (ease optimizer's job)
- There is no other index existing covering the fields from left to right. In that case, the other index will automatically be reused to solve the requests

Remark for Hana DB: please compare to DB guidelines, if creation of DB Index makes sense at all. E.g. when more than 500k entries are expected and when Index shall only be generated for a single column.

Example: Creation of a Secondary Index (1)

Use transaction SE11 to change the database table ZCI_XX_D_ROOT. Click on 'Indexes...'



Example: Creation of a Secondary Index (2)

Add the fields MANDT and INVOICE_ID as index fields. It is OK if the database index is non-unique. Save and activate.

3

Dictionary: Change Index

Index Name: ZCI_01_D_ROOT ID

Short Description: Ir

Last changed: HAEHNER 22.11.2012 Original language: EN English

Status: New Not saved Package: \$TMP

Index does not exist in database system DB6

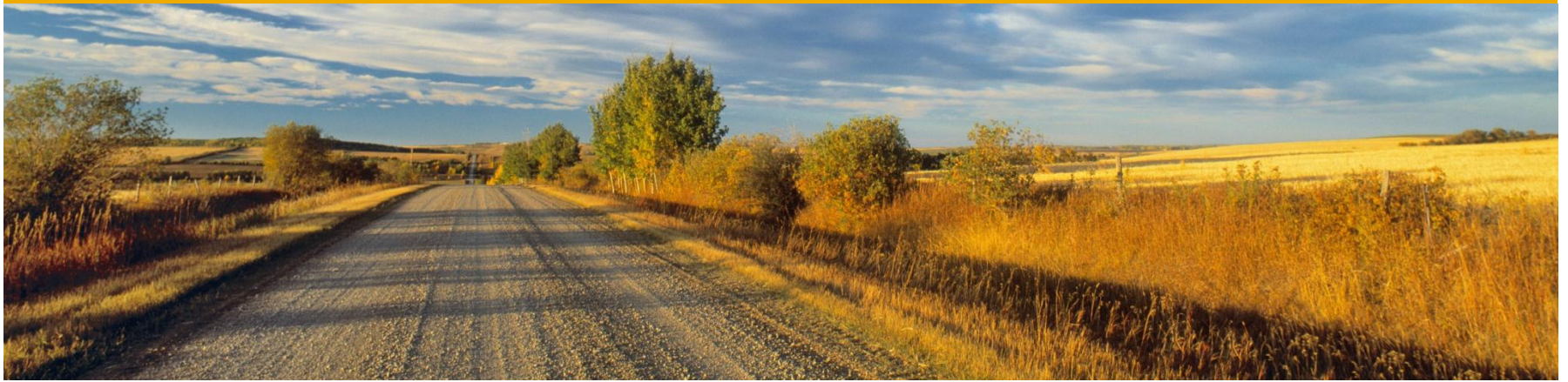
☒ Non-unique index

- ☒ Index on all database systems
- ☐ For selected database systems
- ☐ No database index

☐ Unique Index (database Index required)

Table Fields

Field name	Short Description	DType
MANDT	Client	CLNT
INVOICE_ID	Internal Number	INT4



Determinations

Lazy Determination Execution

Scenario

A time-consuming after modification determination calculates less important fields or fields, whose content is not immediately reflected on the user interface.

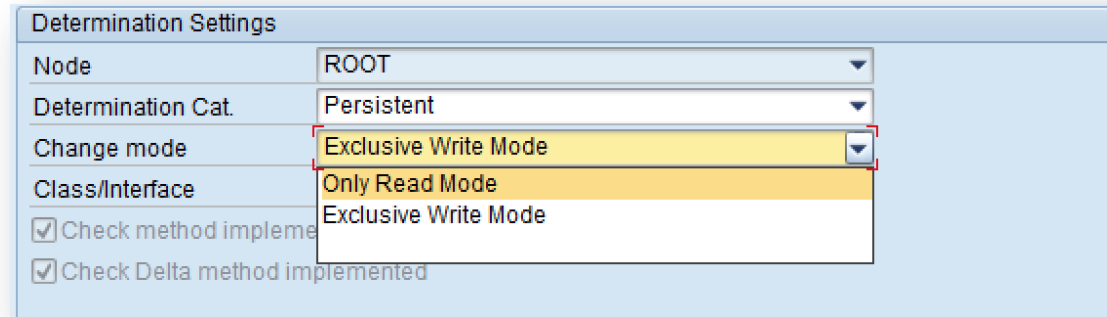
Solutions

Instead of executing the determination immediately after each modification, configure it to the finalize determination time. It will only be executed during the save and thus it will be less often executed than before.

Example

The admin data determination is usually configured to the finalize determination time. Thus the user who changed instances will be only updated during save, but not immediately after the change. This little time frame of inconsistency doesn't really matter.

Determinations using Only Read Mode



Determination Settings	
Node	ROOT
Determination Cat.	Persistent
Change mode	Exclusive Write Mode
Class/Interface	Only Read Mode
<input checked="" type="checkbox"/> Check method implemented	Exclusive Write Mode
<input checked="" type="checkbox"/> Check Delta method implemented	

Scenario

A determination updates only instances of a transient node.

Problem

By default, BOPF locks instances automatically while editing them. As transient node instances are always isolated to the current session scope, it is not needed to lock them.

Solutions

Use the edit mode „Only Read Mode“. It will suppress time consuming framework logic for locking the modified instances and the write node evaluation.

Combination of Determination

Scenario

There are two determinations having the same request nodes and trigger conditions and operating on the same data.

Solutions

It would be possible to combine them. Thereto read the data at the beginning in internal tables, execute both of the determinations' logic and flush the modified data to the buffer at the end. On the one hand, this reduces the internal buffer access costs (especially if many instances are buffered), but on the other, it worsens your application design.

Before Retrieve Determination Configurations

Scenario

A time-consuming “before retrieve” determinations appears more than expected in the performance trace.

Solution

“Before retrieve” determinations can be configured in two different ways. Depending on the configuration, they will be executed multiple times:

- If their request node is equal to their assigned node, BOPF will execute the determination exactly once for each key requested in the RETRIEVE call.
- If their request node is a subnode, BOPF will execute that determination every time the consumer navigates to the subnode via RETRIEVE_BY_ASSOCIATION call.



Queries

Paging

Scenario

Queries fetch data from a business object. If the size of data of a business object increases, the query usage can slow down the scenario performance.

Solutions

A consumer using a query should specify the information request as fine grained as possible. Thereto the `IS_QUERY_OPTIONS` parameter should be used:

- Use the `MAXIMUM_ROWS` component in order to restrict the query result
- Use the `SORTING_OPTIONS` in order to get an already sorted result instead of sorting it on ABAP level
- Use paging (`PAGING_ACTIVE`) and hand over the `START_ROW` or the `START_KEY` of the desired page

If the query is implemented, the implementation must also support that parameters.

Generic Result Type Query

Scenario

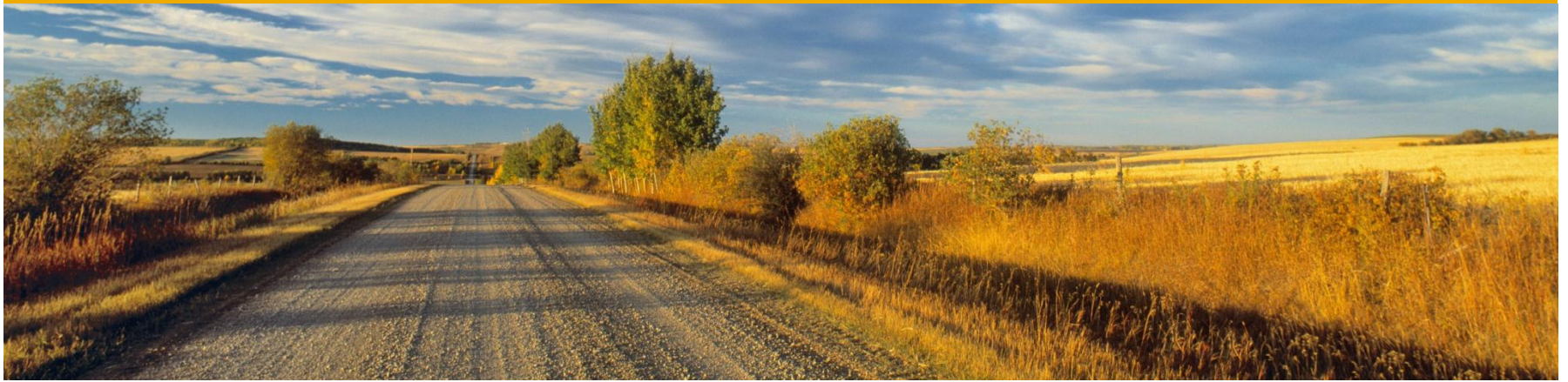
Information contained on different nodes of the business object shall be returned to the consumer by the help of one single core service call.

Problem

Navigating through the business object via RETRIEVE_BY_ASSOCIATION to collect the required data will lead to many database requests.

Solutions

Use a „generic result type“ query and implement a common SQL join statement. If you're using the BOPF authority solution, also mix in the authority checks.



Validations

Lazy Validation Execution

Scenario

A time-consuming validation checks less important fields or fields, which are not immediately reflected on the user interface.

Solutions

Instead of executing the validation immediately after each modification, configure it as action validation on save. It will only be executed during the save and thus it will be less often executed than before.

Validation Sequence

Scenario

Two consistency validations have the same request nodes and triggering condition. The first validation is very time consuming but rarely fails. The second one often fails but is very fast.

Solutions

It would be possible to just execute the fast validation first and in case of an error bypass the second validation. If both validations fail, the user would get just the error message of the first one. If he corrects the issue and executes the check again, the second error message would be visible. The usability would be not perfect but the performance would be much better – especially as the second validation do not fail very often.

- Thus think of combining the validations in a single class and execute the validation logic of the fast validation first. Only in case of no errors, continue executing the slower logic.
- In case of action validations, check the option „Do not execute if errors have occurred“ and maintain the validation dependencies. If an instance fails a validation, it will not be handed over to the follow up validations.

Use Action Validation Usage to check the Save

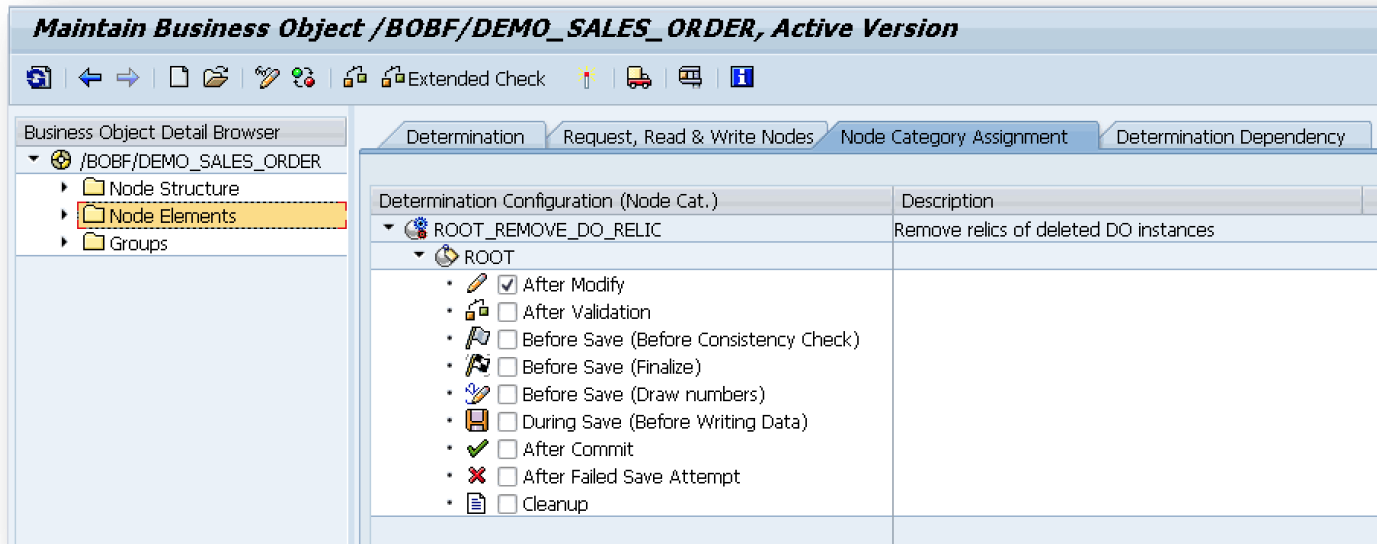
Scenario

A time consuming consistency validation is configured to prevent the save by the help of a save preventing consistency group. In the performance trace, this consistency validation appears very often.

Solutions

Use an action validation configured on „save“ instead of the consistency validation. A consistency validation is always executed, as soon as their condition is fulfilled – even after each modification roundtrip. In addition, save preventing consistency validations needs to have the „check“ trigger maintained. Thus the are additional executed as soon as the consumer executes the „check“/“check and determine“ core service.

Avoid After Validation Time (due to image copy)



Avoid to have a “After Validation” determination in your business object. This will reduce the internal image copy.



Transient Data

Transient Data

Scenario

In many cases data is somehow redundant. For instance, you can derive the IBAN of Germans out of the former account number, postal code and country.

Problem

There are two ways of implementing this in BOPF.

- a. Use a transient attribute “IBAN” which is automatically filled by an after loading determination from the former account number. An after modification determination is needed to keep it in sync.
- b. Use a persistent attribute “IBAN” which is synchronized by the help of an after modification determination.

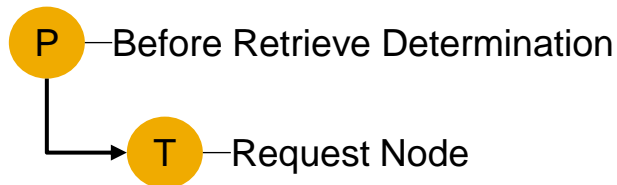
Solution

If a persistent attribute is used, queries and additional consumers (e.g. SADL) can access the attribute. In addition, there is no performance overhead while reading the value – only updating the value will cost performance.

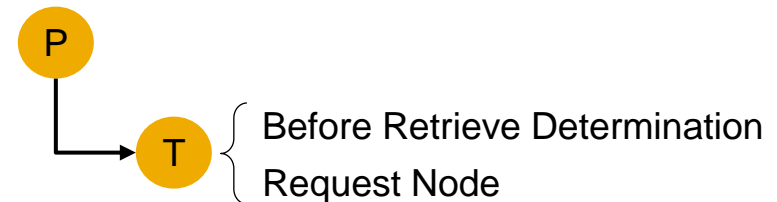
Before Retrieve Determinations

Configuration Alternatives

(a)
**Determination is triggered always by a
“Retrieve-By-Association”**



(b)
**Determination is triggered by “Retrieve”
(but only one time for each key)**



Depending on the configuration of a before retrieve determination, the triggering of the determination is different:

- In case of (a), the retrieve-by-association always triggers the before retrieve determination. The IT_KEY parameter of the RBA call is completely handed over as IT_KEY importing parameter to the determination's implementation.
- In case of (b), a retrieve call executes the before retrieve determination. But only those IT_KEY from the retrieve parameter are handed over as IT_KEY importing parameter to the determination's implementation, which are not requested by a retrieve before.

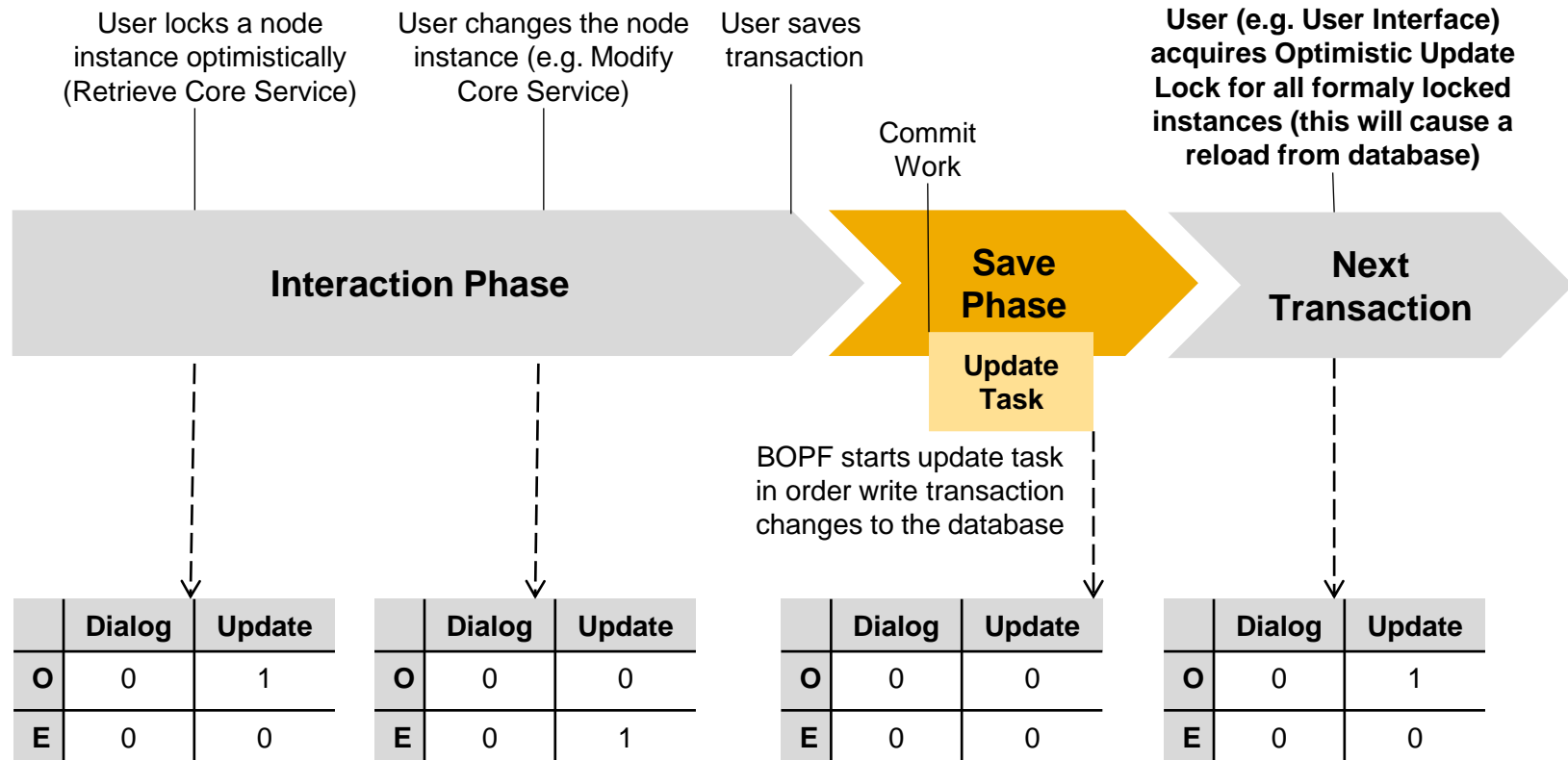


Transactions

Lock Pattern

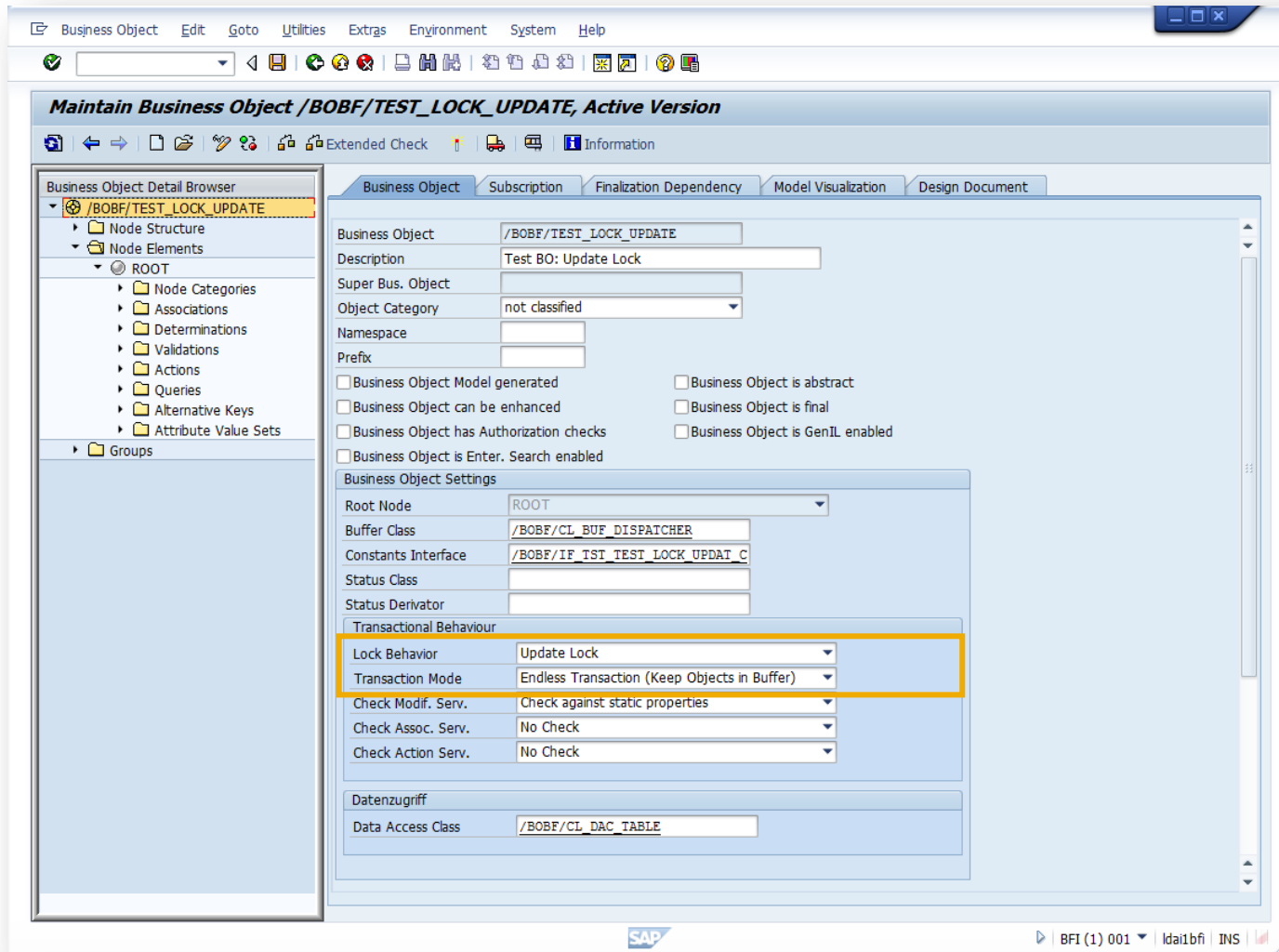
	Update Lock Pattern	Optimistic Dialog and Update Lock	Exclusive Dialog and Update Lock
Description	Only update locks	Keeps optimistic dialog lock during save	Keeps exclusive dialog lock during save
Details	After the save of a transaction the update locks are released due to the processed update task. Thus the user interface has to reacquire all optimistic locks in order to keep the UI fields editable.	<p>If thousands of instances participate at the transaction, the reloading after reacquiring the locks is not possible due to performance reasons.</p> <p>Thus the endless editing lock mode uses optimistic dialog locks to keep informed that only parallel changed instances need to be reloaded.</p>	<p>If thousands of instances participate at the transaction, the reloading after reacquiring the locks is not possible due to performance reasons.</p> <p>Thus the endless editing lock mode uses optimistic dialog locks to keep informed that only parallel changed instances need to be reloaded.</p>
Reload	Yes This reacquiring causes a reload of the data from the persistency.	No	No
Usage	Use this by default.	Mass Data Scenarios, having many instances buffered and less optimistic locks.	Mass Data Scenarios, having many instances buffered and many optimistic locks.

“Update Lock” Pattern Overview

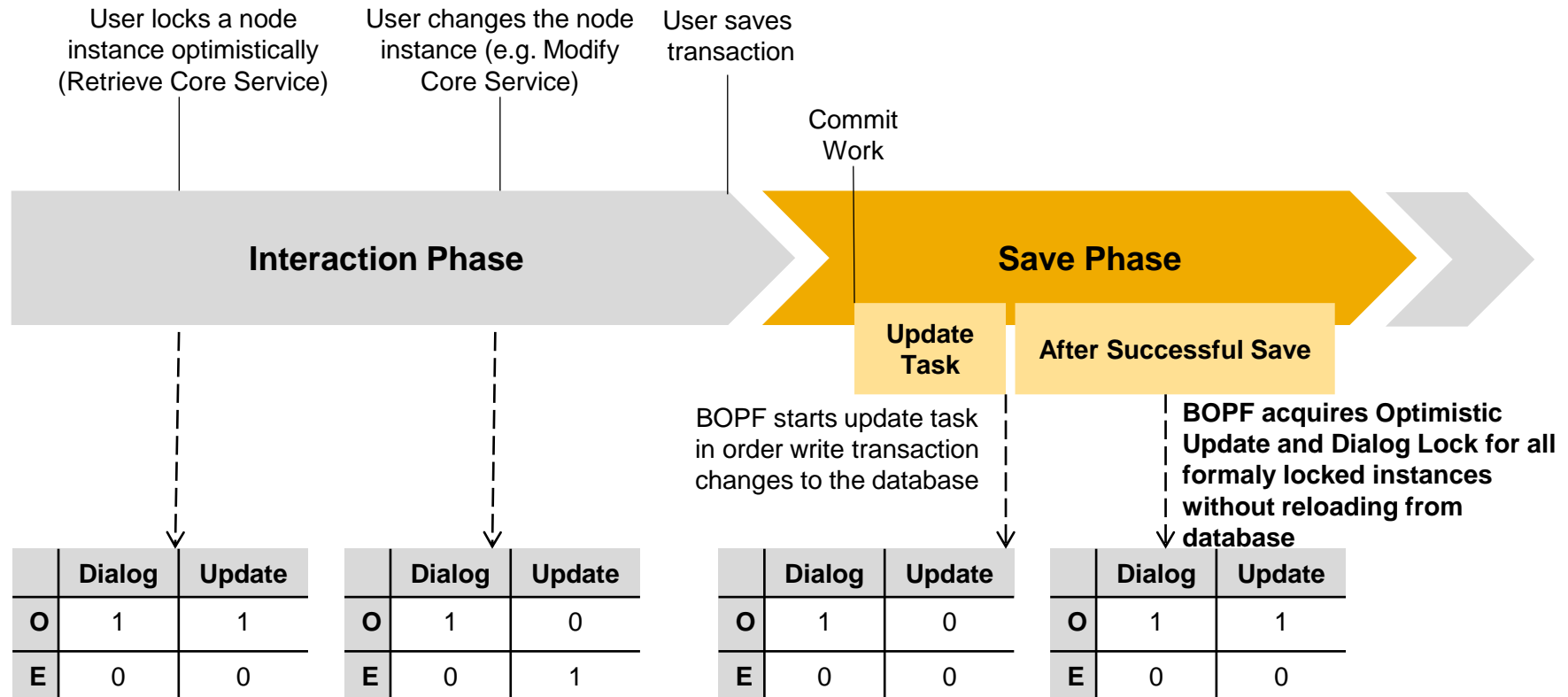


“Update Lock” Pattern

Static Configuration

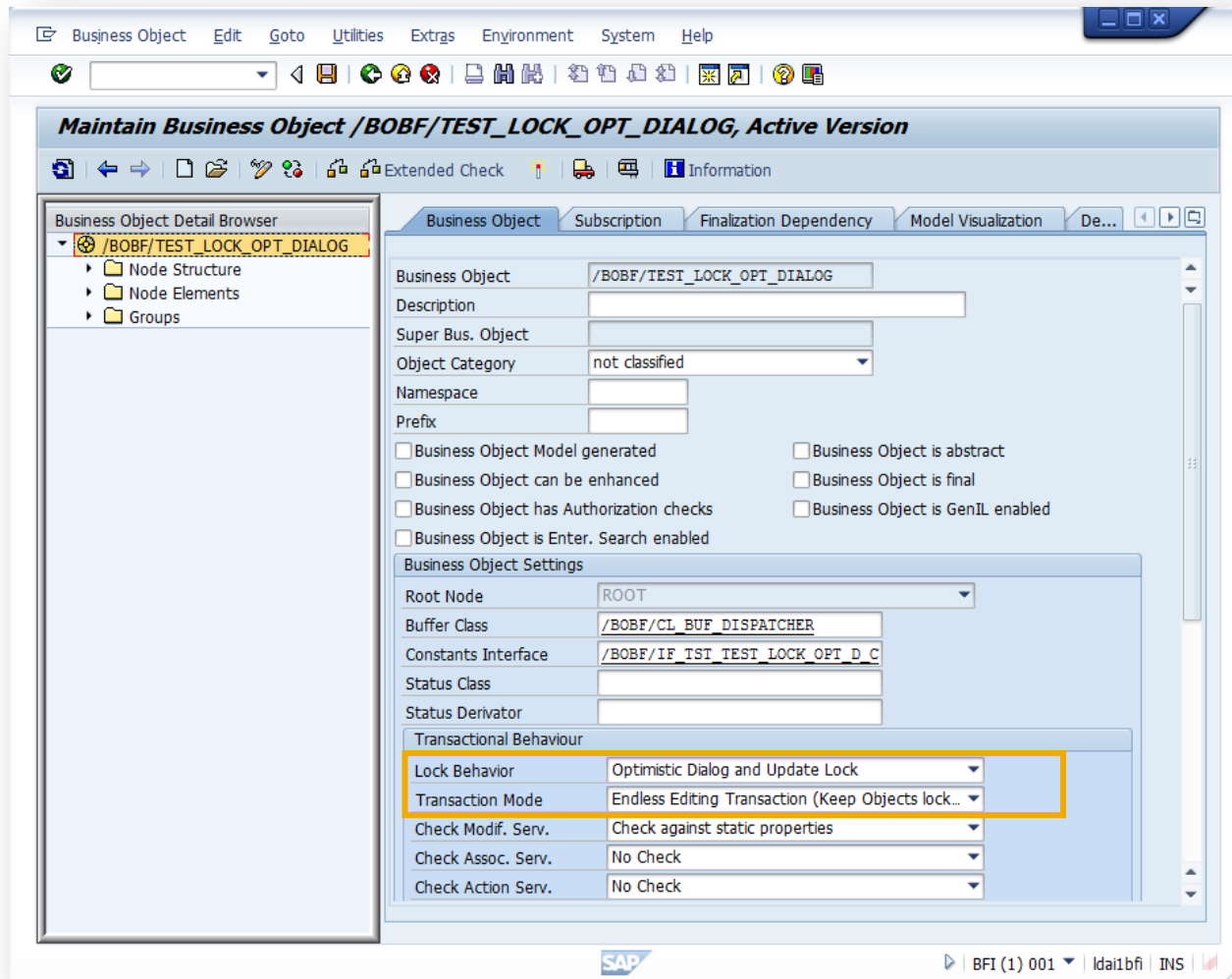


“Optimistic Dialog and Update Lock” Pattern Overview

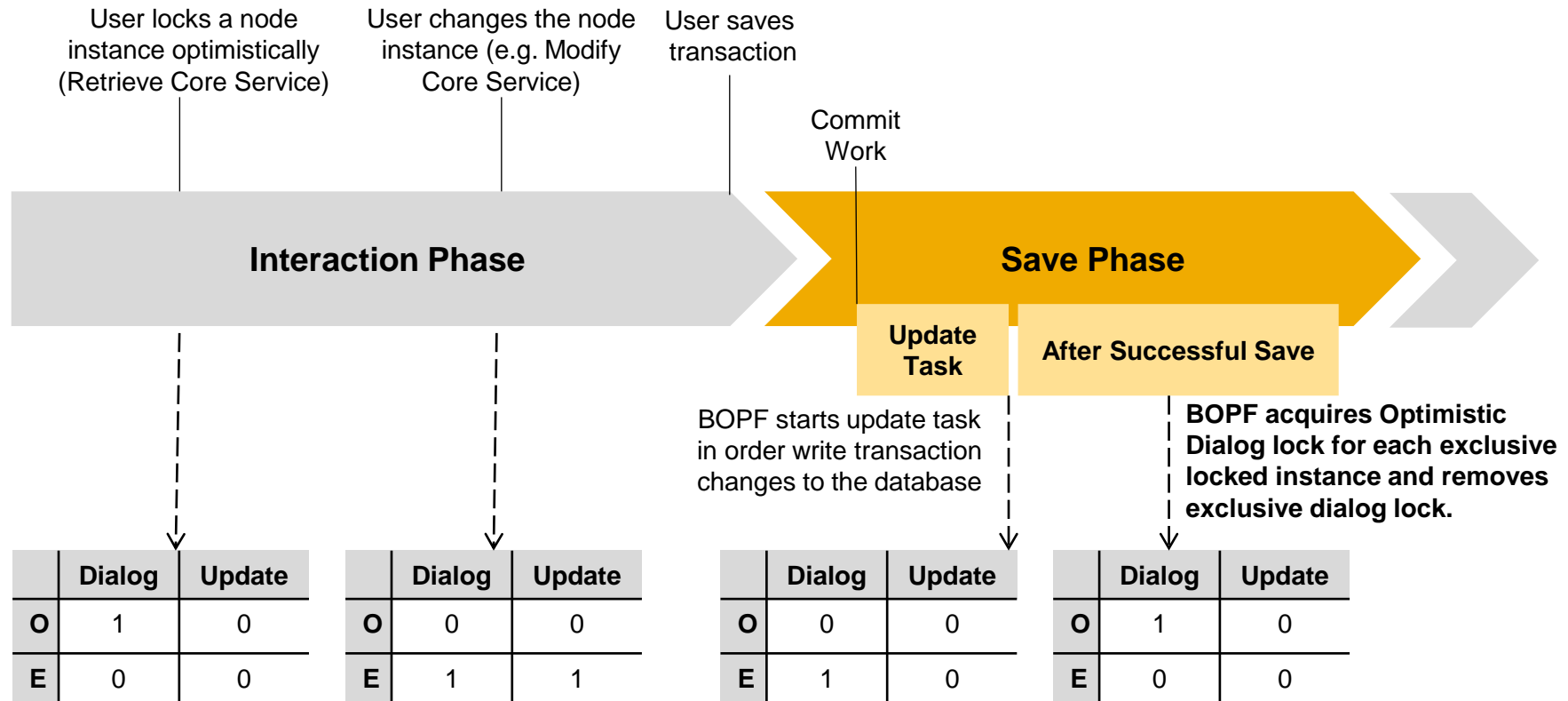


“Optimistic Dialog and Update Lock” Pattern

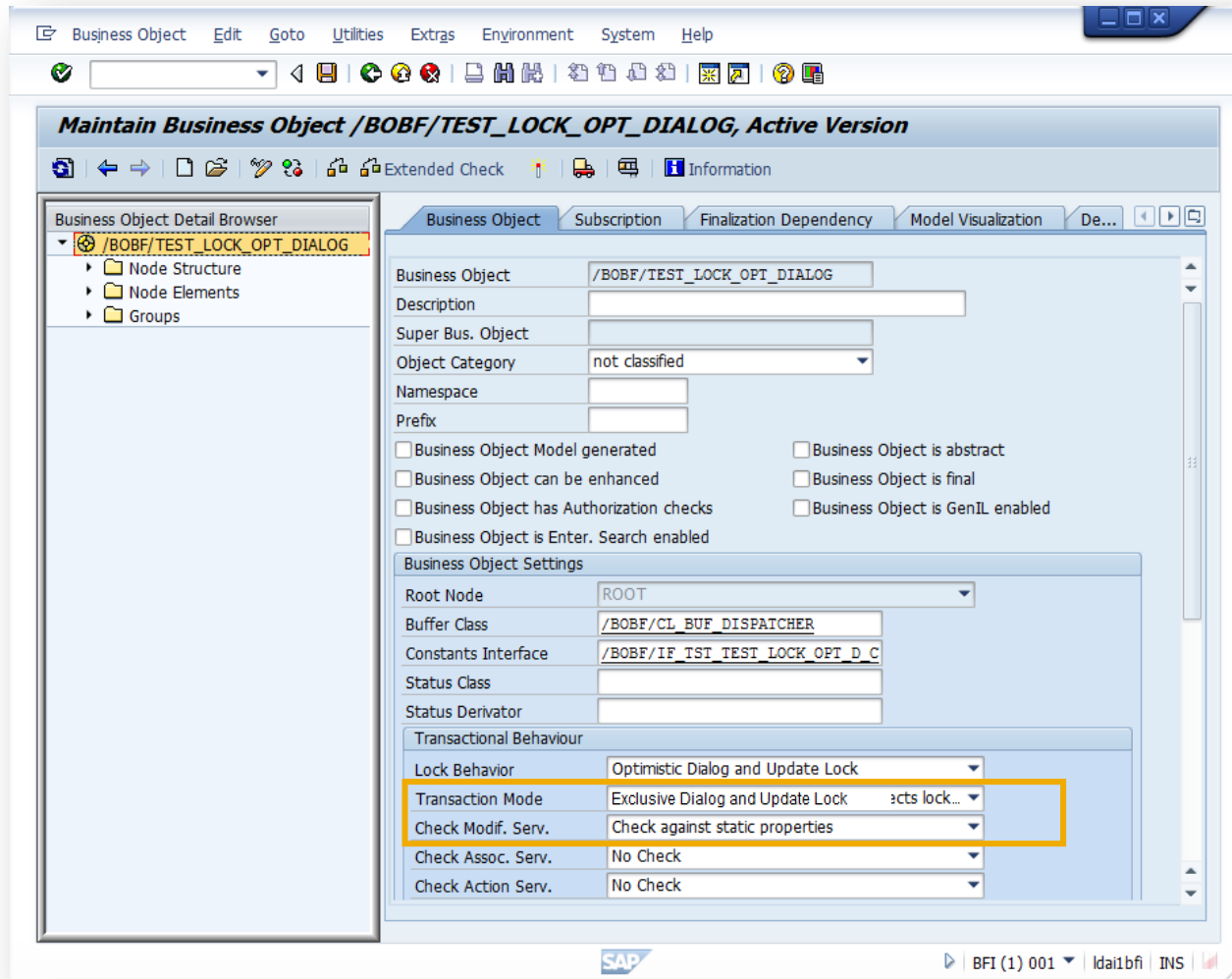
Static Configuration



“Exclusive Dialog and Update Lock” Pattern Overview



“Exclusive Dialog and Update Lock” Pattern Static Configuration



Dynamically overrule Lock Mode

The Lock Mode can be set statically at design time and overruled dynamically at the beginning of a transaction with method

`/BOBF/IF_TRA_TRANSACTION_MGR~SET_TRANSACTION_CONTEXT`

of class `BOBF/CL_TRA_TRANSACTION_MGR` (parameter `IV_ENQUEUE_SCOPE` and `IV_CLEANUP_MODE`).

Attention: Setting the lock mode during a transaction instead of its beginning will lead to a dump.



Properties

Requesting Properties

Scenario

A batch report modifies instances and it is ensured, that this report works correctly. However the properties of the business object are requested and evaluated.

Problem

As soon as a consumer request properties for the first time for a certain node, BOPF will sent further properties and property change notifications automatically during the whole transaction for it. The creation of properties is time consuming.

Solutions

- Don't request properties (via RETRIEVE_PROPERTIES core service) if the consumer does not need them.
- Call cleanup in order to get no further properties and notifications

Reduce Automatic Property Checks

Scenario

BOPF offers the possibility to just return calculated properties to the consumer or to also validate the consumer's requests against that properties. For instance, it is possible that BOPF automatically rejects a modification if there is a readonly property.

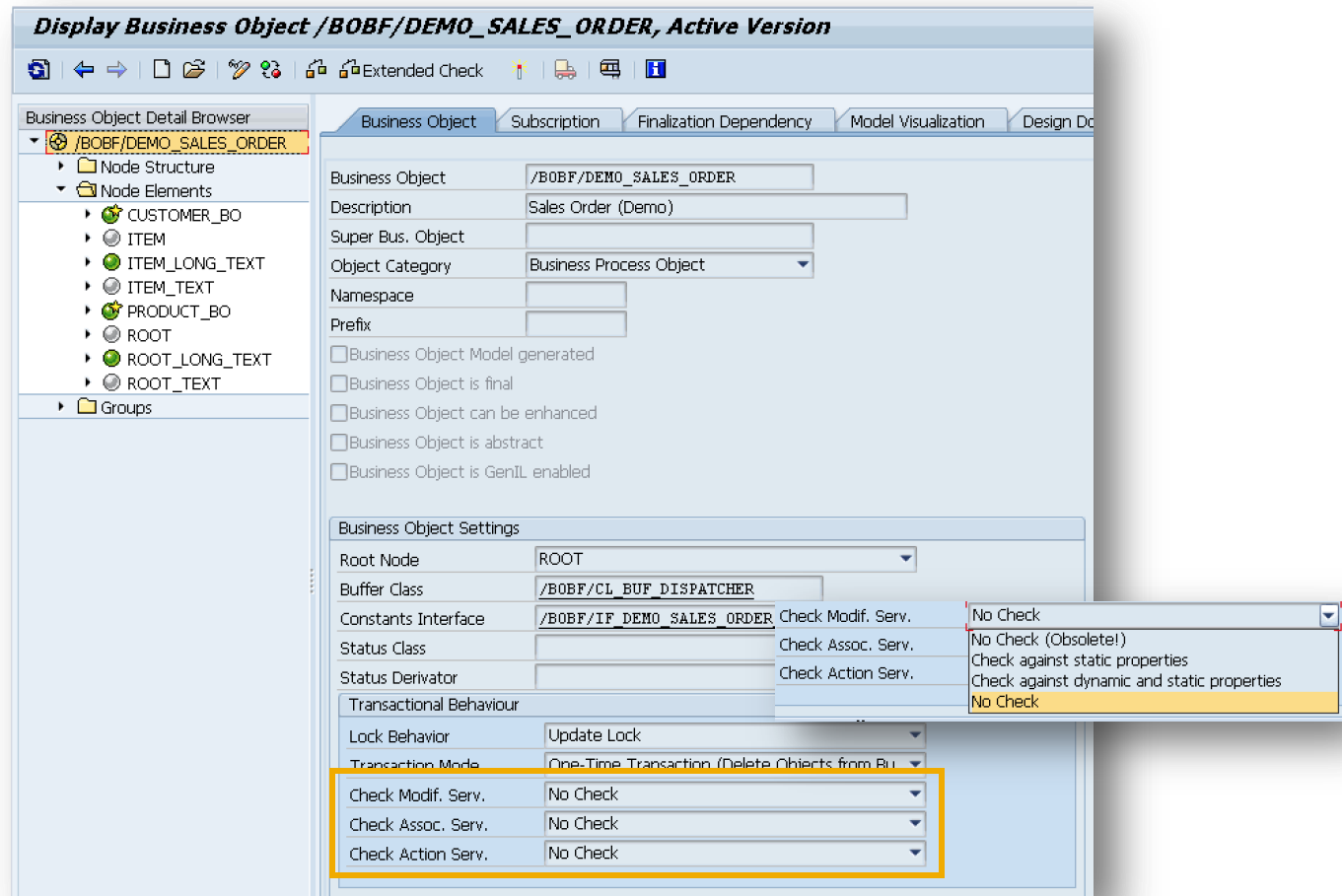
Problem

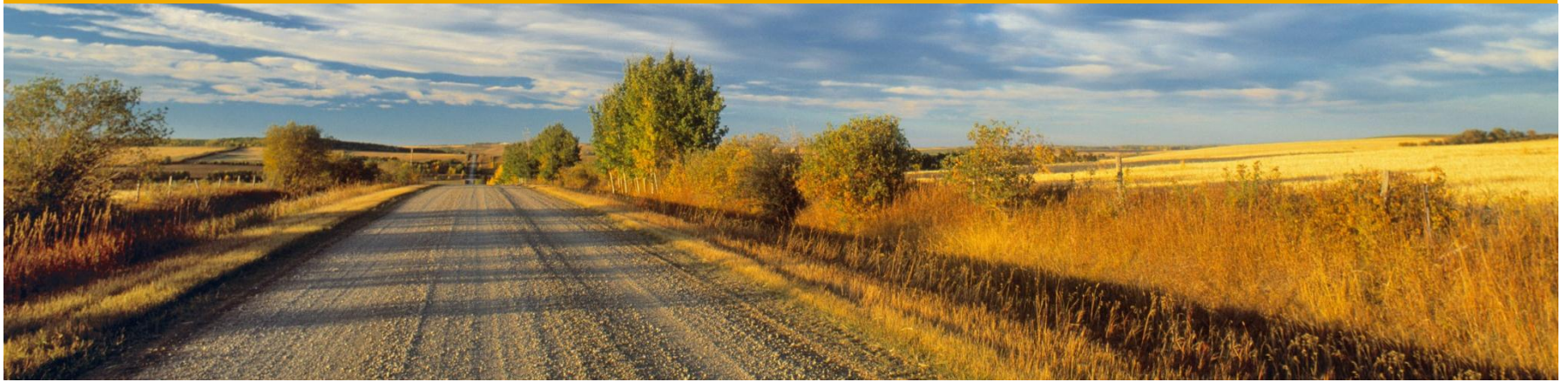
The validation of consumer requests against properties is time consuming.

Solutions

Reduce automatic property checks if the consumer behaves consistent.

Reduce Automatic Property Checks Configuration





Loading Groups

Loading Groups

Scenario

BOPF allows to define loading groups by the flag “separate loadable” on the node level. If a member of a loading group is retrieved, all related instances of the whole loading group is automatically fetched from the database, too.

Problem

If for instance the whole business object is a loading group, the retrieve of one instance will cause the loading of all related instances. This might cause a lot of separate database selects.

Solutions

The loading group feature is deprecate, thus don't use it anymore. If there attributes, which are usually retrieved together, put them on the same node.



Thank you

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, PowerPoint, Silverlight, and Visual Studio are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, z10, z/VM, z/OS, OS/390, zEnterprise, PowerVM, Power Architecture, Power Systems, POWER7, POWER6+, POWER6, POWER, PowerHA, pureScale, PowerPC, BladeCenter, System Storage, Storwize, XIV, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, AIX, Intelligent Miner, WebSphere, Tivoli, Informix, and Smarter Planet are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the United States and other countries.

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

Oracle and Java are registered trademarks of Oracle and its affiliates.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems Inc.

HTML, XML, XHTML, and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Apple, App Store, iBooks, iPad, iPhone, iPhoto, iPod, iTunes, Multi-Touch, Objective-C, Retina, Safari, Siri, and Xcode are trademarks or registered trademarks of Apple Inc.

IOS is a registered trademark of Cisco Systems Inc.

RIM, BlackBerry, BBM, BlackBerry Curve, BlackBerry Bold, BlackBerry Pearl, BlackBerry Torch, BlackBerry Storm, BlackBerry Storm2, BlackBerry PlayBook, and BlackBerry App World are trademarks or registered trademarks of Research In Motion Limited.

Google App Engine, Google Apps, Google Checkout, Google Data API, Google Maps, Google Mobile Ads, Google Mobile Updater, Google Mobile, Google Store, Google Sync, Google Updater, Google Voice, Google Mail, Gmail, YouTube, Dalvik and Android are trademarks or registered trademarks of Google Inc.

INTERMEC is a registered trademark of Intermec Technologies Corporation.

Wi-Fi is a registered trademark of Wi-Fi Alliance.

Bluetooth is a registered trademark of Bluetooth SIG Inc.

Motorola is a registered trademark of Motorola Trademark Holdings LLC.

Computop is a registered trademark of Computop Wirtschaftsinformatik GmbH.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, SAP HANA, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase Inc. Sybase is an SAP company.

Crossgate, m@gic EDDY, B2B 360°, and B2B 360° Services are registered trademarks of Crossgate AG in Germany and other countries. Crossgate is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.