### Change history:

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>April 2007</td>
<td>Original version</td>
</tr>
<tr>
<td>2.0</td>
<td>October 2009</td>
<td>General clarifications and enhancements</td>
</tr>
</tbody>
</table>
# Table of Content

1. Management Summary ......................................................................................... 4
2. Application Life-Cycle Management ................................................................. 5
3. Data Integrity and Transactional Consistency Standards at a Glance .................. 8
   3.1 Data Consistency ......................................................................................... 8
   3.2 Origins of Data Inconsistencies ................................................................. 8
   3.3 Transactional Consistency ......................................................................... 9
   3.4 Definition of “Temporary” or “Permanent” Data Inconsistencies ............... 10
   4.1 Prevention ................................................................................................. 12
   4.1.1 Data Architecture .............................................................................. 12
   4.1.2 Change Management ........................................................................ 13
   4.1.3 Monitoring and appropriate operational procedures ............................ 14
   4.1.4 End-User Training .............................................................................. 14
   4.2 Detection .................................................................................................. 15
   4.3 Evaluation ................................................................................................. 17
   4.3.1 Investigation on Data Inconsistency ..................................................... 17
   4.3.2 Business Continuity ......................................................................... 19
   4.4 Correction ................................................................................................. 21
5. How to Implement the Data Integrity and Transactional Consistency Standard? ... 22
   5.1 Prevention ................................................................................................. 22
   5.2 Detection ................................................................................................. 24
   5.3 Evaluation ............................................................................................... 25
   5.3.1 Business Continuity ........................................................................ 26
   5.4 Correction ............................................................................................... 26
6. How to Measure the Success of the Implementation? ....................................... 28
1 Management Summary

Data integrity and transactional consistency are crucial factors for the success of each SAP solution: Your daily business operation – from the end user to management decisions – relies on correct and up-to-date data being available at the right time.

As a consequence, data inconsistencies can lead to severe costs – e.g. due to lost business deals or a non-availability of your solution.

This white paper explains which measures have to be taken and which preconditions need to be assured in order to avoid and handle inconsistent data in your SAP solution landscape. This paper does not only outline how you should set up your organization and processes in order to reliably detect and delete inconsistencies, it also focuses on the measures you have to take before your actual start of production – data integrity and transactional consistency need to be addressed during the complete lifecycle of an SAP implementation project.

This document outlines the basic concepts SAP recommends to ensure data integrity and transactional consistency. The standard process with its four core steps – Prevention, Monitoring, Business Continuity and Correction - is explained in detail. In addition you will find detailed information about the methodology SAP recommends ensuring that your interfaces and programs are transactional correct to achieve the highest possible standards of integrity. Furthermore, the tools that need to be used are explained in a dedicated chapter. Finally it is also described which roles and functionalities need to be represented in your organization to ensure good prevention, fast detection and appropriate correction of inconsistent data.
2 Application Life-Cycle Management

Companies expect from their IT departments that mission-critical business applications run smoothly, without business disruptions, at low cost, and that they can be adapted easily to new requirements. It is the mission of Application Life-Cycle Management (ALM) to achieve this. SAP’s ALM portfolio consists of processes, tools, services, and best practices, to manage SAP and non-SAP solutions, throughout the entire application life-cycle. For details about the complete portfolio, please refer to http://service.sap.com/alm.

According to the IT infrastructure library (ITIL), the application management life cycle comprises six phases:

- Functional and non-functional requirements are collected and evaluated during the requirements phase.
- In the design phase, the findings from the requirements phase are used to specify how the application or IT operation processes are to function, and which IT applications should be used to map the processes.
- In the build and test phase, a system landscape is set up and configured to implement and test the planned scenarios and processes.
- The deploy phase is the transition from a pre-production environment to production operation.
- The operate phase groups tasks that are performed after system startup, to ensure the availability and stability of the solution. These tasks include activities such as system administration, system monitoring, business process monitoring, message processing (Service Desk), root cause analysis, issue management, and service delivery.
- The optimize phase collects key figures and data from the live solution, to reduce costs or improve performance.

ALM processes span the six phases, to ensure stable operation of the IT solution while enabling accelerated innovation. Optimizing these processes reduces costs and ensures the highest quality of IT operation.

Typically, multiple teams are involved in the ALM processes (see Figure 2.1). They belong to the key organizational areas Business Unit and IT. The names of the organizations differ from company to company, but their functions are equivalent. For example, a program management office communicates business requirements to the IT organization, decides on the financing of development and operations, and ensures that the requirements are implemented. On the technical side, the application management team is in direct contact with the business units. It is responsible for implementing the business requirements and providing support to end users. Business process operation covers the monitoring and support of the business applications, their integration, and the automation of jobs. And SAP technical operation is responsible for the general administration of systems and system diagnostics. Further specialization is possible within these organizations. For example, there may be separate experts for different applications within SAP technical operations, in larger organizations.
Two things are the key to optimizing the collaboration of the groups involved: a common infrastructure, and a clear definition of the collaboration processes, including the activities involved, responsibilities, and service levels. The infrastructure is provided by SAP Solution Manager as a collaboration platform. It provides role-based access to all functions required (provided either by SAP Solution Manager itself or by integrated tools), via work centers. It also provides all related information, centrally, so that all stakeholders involved have easy access to the information they require. Many customers have defined collaboration processes. SAP has leveraged the experience of these customers, and of its own application life-cycle management experts, to create best-practice descriptions of important ALM processes. These documents are published as E2E Solution Operations standards in SAP Service Marketplace at [http://service.sap.com/supportstandards](http://service.sap.com/supportstandards). Customers can refer to these standards when optimizing their own IT processes.

With Run SAP, SAP provides a methodology for the implementation of the End-to-End Solution Operations standards. The road map for Run SAP guides through defining the scope of the operations to be implemented, preparing a detailed plan, doing the setup, and running SAP solutions. Moreover, it helps to find the right strategy and tools to implement ALM. The road map provides not only what needs to be implemented but also information about how it needs to be implemented, in the form of implementation methodology documents and best-practices documents.

Currently, SAP provides the following standards:

- *Solution Documentation and Solution Documentation for Custom Development* define the documentation and reporting required for the customer solution
- *Incident Management* describes the incident resolution process
- *Remote Supportability* contains five basic requirements that have to be met to optimize the supportability of customer solutions
- *Root Cause Analysis* defines how to perform root cause analysis, end-to-end, across support levels and technologies
Exception Handling and Business Process and Interface Monitoring explains how to define a model and procedures to manage exceptions and error situations during daily business operations, and how to monitor and supervise mission-critical business processes.

Job Scheduling Management explains how to manage the planning, scheduling and monitoring of background jobs.

Data Integrity and Transactional Consistency avoids data inconsistencies, and safeguards data synchronization across applications, in distributed system landscapes.

Data Volume Management defines how to manage data growth.

Change Management enables efficient and punctual implementation of changes with minimal risks.

Test Management describes the test management methodology and approach for functional, scenario, integration and technical system tests of SAP-centric solutions.

System Monitoring covers monitoring and reporting of the technical status of IT solutions.

System Administration describes how to administer SAP technology to run a customer solution efficiently.

Custom Code Management describes the basic concepts of custom code operation and optimization.

Security describes basic activities to setup, maintain and evolve security measures for the operation and organization of SAP solutions.

Upgrade guides customers and technology partners through upgrade projects.

Out of this list, this white paper describes the standard for Data Integrity and Transactional Consistency.
3 Data Integrity and Transactional Consistency

3.1 Data Consistency

In the early IT days, data consistency and correctness was given by the fact that there was an application architecture with one system, one disk sub system, and one database. Commit cycles ensured the completeness of transactions at any time. In today’s distributed system landscapes, data integrity cannot be ensured anymore. Different business process “leading” systems require data synchronization across applications; several databases, on disk and in memory, have consistent states within itself, but not across the units; on the sub system level, data is stored across several storage systems. To summarize today’s situation: within distributed system landscapes there is no synchronization point across systems (end-to-end) anymore that ensures data consistency and correctness.

We could speak about two types of consistencies:

- Transactional consistency deals with data consistency within one business system
- End-to-End transactional consistency (application consistency) deals with data consistency between many business transactions targeting one or more applications and systems.

SAP provides best practices and consistency reports/procedures that allow synchronizing the transactional data end-to-end across the different business applications. Depending on customer specific parameters and requirements (like data volume, system resources, 24x7 operations, etc) SAP’s given standards need additional implementation focus to ensure end-to-end transactional consistency and correctness. IT operation has to ensure the ongoing regular check procedures as well as recovery mechanisms to rebuild after a failure situation potentially caused by hardware errors, user error, data manipulation, etc.

3.2 Origins of Data Inconsistencies

Often, the underlying root cause for data inconsistencies in an end-to-end solution landscape is typically one of the following scenarios.

- In one scenario, End Users or Key Users are not aware of manual activities they have to perform in order to guarantee integrity. For example, this could be a change of master data in one component without sending (as a manual procedure) this update to other systems, or entering incorrect data into the system.
- In another typical scenario, an interface is not working properly due to whatever reason. Typically, this is recorded in an error log on one side of the interface and may or may not trigger a workflow to somebody for resolving the issue.
In another main scenario mentioned here, jobs are not running correctly or have not been started at all due to insufficient time. If these jobs should for example load data into a data warehousing solution (BW), the business reports based on these missing data are not accurate and may lead to wrong business decisions.

Inconsistencies between systems may also arise during recovery of systems data, when older data has been restored in one system.

It is vital for solution operations to create 100% transparency with regard to all of these issues. The best practices to address issues with data inconsistencies covers several areas that reach from application design principles, such as guiding the user, and checks to simply avoid inconsistent data entry, up to proper interface, workflow queues and batch jobs monitoring and CIO level reporting on any exceptions and inconsistencies. If the proper “integration monitoring” and reporting is not setup properly, inconsistencies are guaranteed.

### 3.3 Transactional Consistency

As mentioned before transactional consistency plays a major role to ensure data integrity. Here we will explain the basic principles.

To explain how transactional consistency can be achieved by adhering to the “logical unit of work” (LUW)-principle, the concept of database LUWs and SAP LUWs needs to be quickly explained.

A database LUW (DB LUW) or simply database transaction is a non-separable sequence of database operations. At the beginning and end of the LUW, the database is in a consistent state and the LUW is either fully carried out by the database system, or is not carried out at all. It is opened at the start of a transaction (when the connection with the database is made), and whenever a previous DB LUW is closed following a database commit.

The database LUW is closed with a database commit. It is only in the commit that the data is written to the database (after which it can no longer be reversed). Before the database commit, you can undo the changes using a database rollback. Database LUWs allow you to encapsulate logically related actions from a business process.

An SAP logical unit of work (LUW) can be described as the SAP method for mapping several changes into one single DB LUW or as a sequence of dialog steps building one logical unit in the sense of a business process.

Basically, you can say that when programming your interfaces or using a user exit the “all or nothing”-principle should be kept, which means that either all changes are posted to the database in a single LUW or they are not posted at all. In addition it needs to be ensured that appropriate status information is provided to the application whether the data has been successfully committed to the database or not, so that the right restart point can be detected in case of errors without posting data twice or not at all.
3.4 Definition of “Temporary” or “Permanent” Data Inconsistencies

When talking about inconsistencies, it is important to differentiate between the terms *Difference* in contrast to *Inconsistency*. Difference relates to temporary mismatches between data that will always occur in connected running systems (due to the processing times of update tasks, IDocs, BDocs and other interfaces, different scheduling frequencies between systems), thus representing a temporary data inconsistency.

Whereas a permanent inconsistency means a mismatch that does not disappear when all system activities are processed successfully and is thus much more critical from a business point of view. Therefore, the investigation whether an inconsistency or a difference is observed is one of the main tasks before a correction is attempted. This will be discussed in chapter 4.
4 Architecture and Process Flow

Handling of inconsistencies can be divided into 4 phases (see figure 4.1):

- **Prevent**: Prevention of inconsistencies should be taken into account during the process design and training phase already. Transactional consistency needs to be ensured during the program design phase, there should be sufficient End User training before Go-Live to enable your End User to handle even unforeseen exceptions according to defined procedures. Preventing data inconsistencies should also be kept in mind in the business process design phase as well as during the design of customizing and system data change processes.

- **Detect**: Sufficient interface monitoring and corresponding error handling procedures are a prerequisite for data integrity. By implementing and performing an inconsistency monitoring procedure, inconsistencies can be detected at an early stage, so that further harm is prevented.

- **Analyze**: The root cause has to be found in order to prevent further inconsistencies in the future. Further, there should be a business continuity concept so that when an inconsistency is detected, it can be investigated and decided quickly if and to what extent you can continue to work with the system and how to revert back as fast as possible to a normal business operation.
Correct: Correcting data inconsistencies should follow root cause analysis whenever possible, so that the root cause is eliminated before the data inconsistency is corrected by an appropriate method, as well as defined correction procedures for each important, business critical object.

4.1 Prevention

One of the key activities of data integrity management is to avoid data inconsistencies through different measures. In the following we will discuss them.

4.1.1 Data Architecture

No clear definition of a leading system

If different systems contain and use the same master or transactional data within your solution landscape, you should clearly define one leading system for every data object. This means that every change of data is validated in the leading system and is distributed from there to all other systems that use these data objects. Direct data distribution between non-leading systems without the involvement of the leading system may lead to data inconsistencies. Therefore, it has to be avoided in any case!

Which systems are leading for which data object should have been defined in a very early stage of your implementation project – ideally while blueprinting your solution. When defining the leading systems for the different data objects, you also need to define replication procedures for the different data objects.

Please note that another important question that needs to be handled is whether master or transactional data should be able to be changed at all in systems other than the system defined as the leading system.

From a technical point of view, you also have to ensure that distribution of data changes are properly serialized. In case different fields of the same data set are changed in different systems and are replicated, it is important that the chronological sequence of the changes is kept in order to have correct information in all connected systems.

In addition, data belonging together from a logical point of view should be kept in the same leading system and you need to define in detail if complete sets of data should be exchanged between the systems or only delta information should be replicated after a change of data.

If the business process requirements that are addressed to your SAP project are such that it is not possible to define a clear leading system, you should ensure that only distinct data sets are maintained in both systems (for example another material type in both systems or sales data in one system and production data in another) or you should implement a cross system lock. In this case data cannot be changed in a connected system if the affected data set is changed (and locked) in one system of your solution landscape.
4.1.2 Change Management

No Transports into a Running System

Another precaution that needs to be taken in order to prevent data inconsistencies in your productive environment is that you avoid uncontrolled imports of coding transports into your productive system while it is running.

In order to avoid data inconsistencies due to imports into your production environment, you should either transport changes into your system only during downtimes when there is no business activity executed or carefully control the process while understanding possible consequences and taking appropriate measures to avoid them.

No Changes to Existing Customizing Objects

A complete and correct setup of your customizing is an important prerequisite to keep your data consistent over all components of your SAP solution landscape. You have to ensure that this prerequisite is met before the start of your production and thoroughly tested before the Go Live date.

As a general guideline, customizing for existing objects participating in the business processes should never be changed. Therefore, we highly recommend to make the customizing of the productive client “non-changeable” and that you change the maintenance status for tables that need to be maintained regularly only for exceptional cases.

In cases where customizing changes may be required, consider the impact of your changes on further processing and usage of existing documents and use appropriate correction reports if needed. Do not forget to test these reports before running them in your productive environment and work closely together with the Business Department in order to prevent undesired results!

In general, customizing changes should be included in the change management process. Keep in mind that deep knowledge of the application’s customizing as well as of the intended business process using these customizing entries is needed when you intend to change your customizing entries.

If changes in customizing have to be done, it is important to consider how those changes will affect already existing business objects and implemented processes.

Incorrect Programming

Programming errors can lead to data inconsistencies where the mismatch between two data sets could be purely technical (e.g. an incorrect sign in some formula) or logically if the data calculation rules are incorrect (e.g. using the document date instead of a posting date) somewhere.
Another example of incorrect programming leading to inconsistencies are mistakes where the logical unit of work (LUW) concept has been neglected (see previous chapter).

In order to avoid the above root causes of data inconsistency, it is important to implement corresponding application lifecycle management processes (see chapter 5.1).

### 4.1.3 Monitoring and appropriate operational procedures

#### Setup Monitoring

One of the key activities to prevent data inconsistencies is a monitoring process that is set up covering all relevant components and process steps, can generate promptly alerts, and contains detailed work instructions and clear responsibilities in case of an incident or error. See the next process step "Detection", which deals with monitoring issues.

#### Incorrect Error Handling

In the case of incorrect error handling, every involved programming is correct but the operation of the system has lead to errors between systems or data within a system. This situation is most common for business processes spanning two systems.

Typical scenarios are missing data in one of the two systems caused by IDocs that are in error state and are not reprocessed in a timely fashion. Another common scenario would be to overwrite newer data by incorrect reprocessing of an old IDoc. A good indicator for such a case would be a large number of old erroneous interface data – for example IDocs or BDocs – in the system.

Trained operational staff and implemented administration standards are key in order to avoid incorrect error handling. For more details see SAP operational standard for System Administration.

### 4.1.4 End-User Training

Data inconsistencies may also occur due to an entry of incorrect data by end users. This may be due to a non intuitive user interface misleading the end user, missing or incorrect master data, missing definition of workaround and work instructions for non standard situations, incomplete training, for example if the end user does not know how to handle exceptions, or normal human failure.

Most often, this root cause leads to inconsistencies between the real world and the information stored in the system, but it can also lead to inconsistencies between systems.

Basically, you should ensure the following:

- End users are sufficiently trained to be able to perform the business process steps (e.g. in an application) completely and in the correct order (for example, period end closing and period end closing reconciliation)
End users need to be trained in handling any exceptions that might occur.
End users are aware of dependencies between the different data. Therefore, information about the underlying data model should be incorporated in the end user training sessions and documentation.

Another aspect of faulty user input are too extensive authorizations – e.g. users should not be allowed to change field input via debugging in a productive system.

## 4.2 Detection

Detect problem situations as early as possible in order to solve them as fast as possible - before they become critical for the business. Enable the customer’s support organization to respond to and to solve problems more proactively.

**Interface Monitoring**

Since one of the main origins of data inconsistencies are interfaces, you have to ensure that all interfaces in your SAP solution landscape are constantly monitored to ensure a timely reaction to any unforeseen incidents. Since a data inconsistency is not an isolated event in your business process and can have consequences on the data quality of subsequent business process steps, the impact of inconsistencies is the larger, the longer no appropriate actions are taken. So the observed inconsistencies need to be corrected and their root cause needs to be eliminated timely. Appropriate interface monitoring needs to be set up for all key interfaces within your SAP solution landscape – no matter whether they connect SAP components or non-SAP components.

**Data Integrity Monitoring**

Furthermore you have to be aware that data inconsistencies have not necessarily to be caused by erroneous interface communication, but can also have other causes, such as not executed background jobs or errors in a transaction or program.

Apart of proactively monitoring the possible root cause such as failed communication, aborted jobs, or program errors, data inconsistency detection should also be a topic within your business process monitoring procedures. SAP offers various data integrity reports for different applications and business objects. In case those business objects have been modified, the reports have to be adapted or extended to cover new customer data structure. Those reports have to be executed regularly in order to be able to react as quickly as possible.

**Application-specific Monitoring on Business Objects**

Further applications specific monitoring of business objects allows discovering issues with consistency on application level which appear if one step in an end-to-end process has failed or has not been executed. For example failed goods movement postings represent an inconsistency between the real world stock information and the book inventory. This business error can be monitored using Business Process Monitoring.
Monitoring Process

The team responsible for interface monitoring should monitor the availability of the interfaces that are relevant for your business process so that unavailability is detected timely. In addition, the performance and throughput needs to be monitored so that business processes that rely on a fast communication between systems can run at their required level.

While setting up your interface monitoring, you need to define monitoring objects as well as detailed monitoring and error handling procedures. Here you should keep in mind which priorities certain interfaces have for the different business processes. The responsibilities for the different tasks need to be clearly defined and communicated within your organization and the required detailed documentation needs to be available for all involved groups.

![Diagram of Inconsistency Monitoring Process](image)

**Figure 4.2: Inconsistency Monitoring Process**

Data inconsistency monitoring should be performed regularly by running the corresponding inconsistency reports for the most critical business objects and monitoring the results by the business process operations team. Check if the inconsistency reports can be run in a way that they do not show temporary inconsistencies. If this is not possible you can schedule the reports twice and filter out temporary inconsistencies by comparing the results.

If an inconsistency is detected, a root cause analysis should be performed. The root cause analysis should still check if there is a real technical inconsistency and not just a temporary one. Inconsistent data should be corrected after the root cause has been eliminated.

Inconsistencies may cause further issues with business processes handling dependent data. By implementing and performing an inconsistency monitoring procedure, inconsistencies can be detected at an early stage, so that further harm can be prevented.
Detailed information about how to set up and execute your interface, business process and data integrity monitoring can be found in the respective SAP operational standard for Business Process Monitoring, so here we did only summarize the main points you should focus on. See chapter 5 for further implementation-related topics.

4.3 Evaluation

The goal of any procedure regarding inconsistencies should be two-fold: The procedure should identify the inconsistent data including dependent data and the root cause of the inconsistency, as well as determine the business impact and make the decision whether a productive use of the system should be continued or disrupted.

4.3.1 Investigation on Data Inconsistency

The general procedure described in this chapter should provide a starting point and a guideline for a detailed analysis independent on the individual systems and processes involved. The recommended procedure for inconsistency correction is described in the respective chapter of this document.

When an inconsistency is discovered, it is common that it is reported on a very basic level which does not contain very detailed information regarding the technical data or core business processes involved.

Determine whether there is a permanent Inconsistency

At this stage, it is frequently not determined whether there are any “real” inconsistencies in the system or whether only temporary differences have been observed. As already defined in chapter 3, temporary differences could for example be caused by pending IDocs or update tasks and are resolved once all pending system activities are completed. Permanent inconsistencies will remain after the activities are finished.

To judge whether the reported incident is a temporary or permanent inconsistency, you should analyze the data flow “around” the affected data and check whether middleware components work properly, related interfaces are up and running or a backlog exists in an interface transferring the data etc. If this is not the case, please correct the detected error and check whether the assumed inconsistency still persists.

Detailed Information about System Landscape and Business Processes

If it turns out that the observed incident is due to a permanent inconsistency, the next step should be to reach a detailed understanding of the involved business processes, monitoring and reporting activities, and technical objects leading to the observed difference.

While at this stage not all core business processes of the solution are important, it is essential to know in which business and technical data the inconsistency has been observed - especially for the inconsistency detection and the business process steps handling the inconsistent data.
In addition, a very detailed, technical view needs to be obtained on data origin and use of the data by the processes which are in close vicinity to the reported inconsistency as well as any error detection and handling around these steps. The goal is to understand the steps leading to the detection of the inconsistencies and to map corresponding steps of data derivation in case different business processes are involved.

Related to this is the requirement to obtain detailed information about the different landscape components in which the inconsistency occurred or that the affected business process is running in. In this context, it is also crucial to obtain detailed knowledge about the technical relationship between the affected system components – e.g. which system is the leading system for the affected data object etc.

At the end of this process step, the business process context and possible consequences from a technical and business point of view should be clearly visible.

Origin of the Inconsistent Data

Once the related business processes and corresponding system architecture has been analyzed, it is important to identify the mapping of the original data sets in different applications between which the inconsistency originally appeared.

At this point in time, check reports are needed to verify the connected data sets. It is important that temporary differences have been filtered out during the use of check reports by using a time window where no system activities are executed affecting the involved data. Filtering out inconsistencies means that all update tasks or interface processing for the relevant data should be finished and that no new data is created during the time of the analysis. If such a time window is not available, the analysis should be repeated and results be compared between different runs of the same consistency analysis. Temporary differences will disappear between runs of the check reports/tools while permanent inconsistencies will stay in the results between all runs of the same report.

Non-reproducible inconsistencies

Sometimes, the situation arises that inconsistencies exist in "old" data, which means that if the business process steps that produced that data are executed again, the same inconsistencies will not reoccur. If this is the case, the old data should be corrected before attempting to identify the root cause of the inconsistency. This should be done by reloading the data or executing correction reports if possible.

If tests show that the inconsistency does not reoccur, thus affects only "old" data no further root cause analysis needs to be done. It only has to be reassured that all inconsistencies (also the ones that may have been caused by the “original” inconsistency in preceding business process steps) are completely corrected. Please turn the respective chapter of this document for further advice.
Identification of the root cause

If the analysis has revealed so far that the inconsistencies are permanent and do not only affect "old" data, a root cause analysis has to be conducted. The root causes that can lead to inconsistent data are numerous and highly depend on the specific setup of your SAP solution landscape – therefore in the following paragraphs only the most common root causes for inconsistencies are briefly outlined.

The general rule for an inconsistency root cause analysis is not to dig straight into a technical analysis but first to rule out the more operational root causes unless the inconsistency is reproducible. As a first step it should be checked whether one of the measures described in chapter 3.6 “Prevention” can be named as a root cause:

- Transports into running system
- Changes on existing custom objects
- Incorrect error handling
- System misuse
- No clear leading system
- Faulty user input

If these causes can be excluded, the following should be checked: Using appropriate means like check reports, the process chain should be investigated from bottom to top until the first real occurrence of inconsistencies has been identified. Once an inconsistency between two logical connected steps has been found, the technical process involved in these steps need to be debugged and investigated to identify the technical root cause like transactional incorrect interfaces, technical programming error or logical error.

For more information on how to analyze and handle inconsistencies please refer to Best Practices:

"Business Process Monitoring of Data Consistency between SAP BI and Source Systems"

"Data Consistency Check for Logistics"

4.3.2 Business Continuity

When data inconsistencies have been detected, one of the most important decisions to be made is whether it is possible to continue working with the system or not; and if possible, to what extent (to answer the question how severe are business processes affected by the inconsistent data)? Furthermore, it is crucial to ensure a fast return to normal operations if the business has to be disrupted. Business continuity handles exactly these questions.
An essential part of defining the best approach to continue operation is understanding the design of the business processes that caused the observed inconsistency as well as processes that can be affected by the inconsistency. During this business process analysis a detailed description of the end-to-end process across all systems and applications should be recorded. This information, along with the analysis how severe the business impact is, should be the starting point for the decision regarding business continuity.

Very often contacting SAP Support is essential to avoid even more serious business impact by the productive work in the system with inconsistent data.

If the decision is made that certain business processes have to be shut down in the productive system, a fall-back scenario should be activated. A fall-back scenario is a set of procedures and processes that have to be used by the business users in case the productive system is not available for the time exceeding maximum allowed downtime defined in the SLA. Having a clearly defined fall-back scenario described with the necessary level of detail is a must for any implemented SAP solution.

For more information regarding Business Continuity, please refer to following Best Practices:

“Emergency Handling for Recovery of SAP System Landscapes”

“Business Continuity Management for SAP System Landscapes”
4.4 Correction

Once the root cause has been identified, it needs to be corrected. The appropriate measure to do so depends on the nature of the root cause. If a coding error has been identified as the root cause, the coding needs to be corrected. On the other hand, if the issue was caused by the operation of the solution (e.g. incorrect error handling), appropriate measures would be, for example, to improve the administration procedures or to develop new operational procedures.

After the root cause has been identified and corrected, it needs to be investigated how dependent data is affected. This could be either consolidated reporting data or follow-up errors like incorrectly created documents.

Several possibilities exist to recover or correct lost and incorrect data sets:

- Complete Restore of a Backup / Point-in-time recovery
- Restore into a parallel system and reload of data from the parallel system
- Reload data from a leading system
- Correction tools and recovery by relationships to other data or redundant stored data
- Manual correction
- Restore or point-in-time recovery could be also considered. This should not be a first choice and should be applied very carefully, since this may cause other inconsistencies.

Very often, a combination of data recovery methods and tools is required, for example individual incorrect sales documents could be corrected manually on the data base and dependent data be corrected afterwards by correction reports. Each of the different methods has certain advantages and disadvantages leading to different use cases.

Important questions that influence the choice of recovery method are:

- Does dependent data exist for the inconsistent/lost data?
- Could these be used for a reconstruction of data?
- How many Business Objects and how many instances are affected?
- Which quantity/complexity of objects are affected?
- Is a backup available for a point-in-time recovery?
- How much time would the different methods require?

Different methods are evaluated in detail in the Best Practice “Data Consistency Check for Logistics”.
5 How to Implement the Data Integrity and Transactional Consistency Standard?

The implementation of Transactional Consistency & Data Integrity operational standard is described in the Implementation Methodology within Application Management work package of Run SAP Roadmap (see http://service.sap.com/runsap) and can be divided into three phases:

**Design:** The section includes the definition of the goal and the requirements, the process description, the roles and organization, and tools that support you during the later phases, and proposes possible key performance indicators (KPIs).

**Setup:** The section describes what needs to be done during the implementation itself. The first part of this section contains the implementation of the monitoring that is necessary to ensure correct and up-to-date data. The second part describes the setup of the Business Continuity Plan. The following section describes how to setup monitoring of the KPIs.

**Operations and Optimizations:** The section describes the necessary documentation in daily operations, the activities that need to be done at certain points in time, including the implementation of the Business Continuity Plan and the emergency handling for business continuity, and the relevance of the proposed KPIs to measure the success of the implementation of the methodology.

5.1 Prevention

Prevention of inconsistencies has to be tackled already during the process design phase in order to build corresponding data architecture and applications which ensure transactional correctness and avoid data inconsistencies.

There should be sufficient End User training before Go-Live to enable your End User to handle exceptions according to defined procedures. Preventing data inconsistencies has also be kept in mind during business process design phase as well as during the design of customizing and system data change processes.

Main principles how to prevent data inconsistencies are outlined in chapter 4: on the one hand implemented change management process and on the other hand implemented interface and business process monitoring.

Change management is an important topic when it comes to prevent data inconsistencies. To adhere to the guidelines outlined in this document, you need to ensure that proper change management procedures are set up that include sufficient tests of your developments and your customizing entries. In addition, procedures for coding transports between the different parts of your system landscape have to be clearly defined.
Here your custom development team is highly involved – as well as your application management team. End and key users participate in this process as testers of any changes which are to be applied to the system.

The tools that are involved here are general tools to development programs and reports as well as tools for testing and transporting these developments between components of your system landscape.

A detailed methodology and process description is described in SAP operational standards for solution operations:

- **Custom Code Management** which covers the basic concepts of custom code operations and optimizations
- **Change Request Management** enables efficient and risk minimized implementation of changes
- **Change Control Management** deals with the deployment and the analysis of changes within a landscape
- **Test Management** which tackles methodology and approach for functional, scenario, integration and technical system tests of SAP-centric Solutions

Another key element in preventing data inconsistencies is to set up of promptly monitoring. In order to do so, you first have to define monitoring objects and appropriate procedures to monitor these objects. Based on this, detailed error handling procedures have to be worked out that also include key performance indicators (KPIs) based on which appropriate actions are taken. Furthermore, detailed priorities need to be assigned to each of these elements.

The monitoring activities affect all parties within your project organization. The business unit (for example, the business process champion and key users) need to help to define monitoring objects from a business point of view and need to set priorities depending on the business criticality of possible errors related to these monitoring objects. The application management team and business process operations team are the groups that are responsible for executing the monitoring. They are also the people that execute the defined error handling procedures.

Which tools are required for interface and business process monitoring depends highly on the type of components deployed in your SAP solution landscape.

A detailed methodology for interface and business process monitoring is described in detail in the SAP standard for Business Process Monitoring.
5.2 Detection

Figure 5.1: Implement Monitoring Process

Monitoring objects and monitoring procedures should be defined jointly between your business department, your application management and business process operation teams.

After the blueprinting and during the technical development of your solution, it is crucial that you implement an inconsistency monitoring concept. Therefore, it needs to be defined which inconsistency reports should be scheduled, run and monitored regularly. This should be done for the most important business objects which mostly endanger the smooth and reliable flow of your core business processes if they are not in a consistent state. If necessary, such inconsistency reports still need to be created e.g. if custom or modified business objects are used.

You should define appropriate variants for each consistency report you wish to monitor to distinguish between different recipients and target groups. Error handling procedures for inconsistency as well as thresholds for alert triggering should be defined and documented. There should be a scheduling plan for the inconsistency reports, which also considers other system activities because of the system load and to prevent that temporary differences are shown, if possible. Monitoring activities as well as the responsible persons should be defined and the defined monitoring should be included into the service level management concept.
The monitoring procedures also to be thoroughly tested and have to fulfill the End-to-End-approach, which means that they cover all involved components and also contain detailed instructions how to detect data inconsistencies, how to correct them and how to find and solve their root cause. These procedures need to be clearly documented in an operation handbook that needs to be available for every involved employee. All employees need to be trained sufficiently in these processes and escalation procedures need to be clearly defined.

The design of the inconsistency monitoring process should be done together with business process champions, because detailed knowledge about critical inconsistencies as well as the involved business processes is required, whereas the implementation and operation of the monitoring process can be done by the business process operations team.

Several application specific consistency reports/ tools exists which are available per SAP notes as well as included in the SAP standard applications.

SAP Solution Manager provides a dedicated Workcenter “Business Process Operations” which includes dedicated areas for business process and interface monitoring as well as data consistency management. The latter provides a separate inbox where all alerts relevant to data integrity are filtered. The relevant data consistency tools are proposed for investigating and correcting data inconsistencies.

If support is needed for identifying data inconsistencies, there is the possibility to order a data consistency check service.

5.3 Evaluation

General guidelines regarding the methodology to be applied for analyzing data inconsistencies can be found in chapter 4 of this document. It is important that you adhere to the recommended process and consider all topics addressed in that chapter.

The people that help to put this process into practice are in general all parties involved in your SAP implementation, since depending on the business process step different groups are involved.

The analysis whether an inconsistency is temporary or permanent is a task that involves the business process operations team, whereas end users and key users are involved when it comes to understanding the business process “around” the detected inconsistency. To learn more about the underlying technical infrastructure and its interdependencies also your SAP technical operations group or even your IT infrastructure group might be involved. Tasks like determining whether the inconsistencies occur only in “old”, or also in “new” data lie within the area of responsibility of your business process operations team. The final root cause analysis can also involve your development group. Also, first and second level application support teams play a key role in the described process.

The process of an inconsistency analysis involves your monitoring tools for determining whether a “pending” system activity causes the inconsistencies, and check reports to find out whether and where inconsistencies persist. Debugging and development tools are required for a deeper analysis. All these tools highly depend on the specific components, on the setup of your SAP solution landscape, and also on the non-SAP components deployed.
5.3.1 Business Continuity

Several roles are involved in the business continuity process after the data inconsistency has been detected. Key users deliver an important input for the identification of the business impact. Application support employees with the understanding of the technical background also provide important information regarding the severity of the inconsistency to the business process champion. In addition, application management department is initiating the communication with SAP Support. The business process champion is making the decision for the business continuity using the input from the key users, application management department, SAP Support and based on his understanding of the business process flow.

The implementation of Business Continuity Strategy is described in the Implementation Methodology within Application Management work package of Run SAP Roadmap (see http://service.sap.com/runsap).

5.4 Correction

Figure 5.2 shows a roadmap for the correction of inconsistent data.

The described guidelines for the root cause analysis and correction of detected inconsistencies should be included in the implementation of the inconsistency monitoring process as general error handling procedures and made available for the involved people.
The business process operations team, which is responsible for performing the root cause analysis as well as the correction of the inconsistency, should be trained with the content of these general guidelines.

The same is true for error handling procedures for specific inconsistency cases.

Before proceeding with the correction of inconsistencies, the following factors should be considered:

1. Before starting with the correction of data inconsistencies, make sure that real inconsistencies and not temporary differences are observed.

2. During the correction process, not only originally detected inconsistencies should be fixed, but also the inconsistencies or erroneous data resulting from the original inconsistencies. This could be either consolidated reporting data or follow-up errors like incorrectly created documents. To understand the impact of the inconsistency on depending data, the business processes using the original data have to be followed. If follow-up documents are created (for example controlling or financial data for logistics data), these need to be corrected as well using appropriate actions.

3. Before proceeding with the repair of inconsistent data, an investigation is required to determine if new or more recent (up-to-date) data exist in the system. If possible, these data should not be overwritten with the correction process.

After correction, a review of the documentation and the procedures is needed in order to verify and optimize monitoring, analyzing and correction procedures.

Further follow up activities on raised incidents and documentation of taken measures will avoid appearance of inconsistencies in the future.

A lot of application specific consistency reports/tools are partly available per SAP notes - which can be find on Service Market Place – or partly available per standard in the system for the correction of inconsistencies.

The SAP Solution Manager workcenter “Business Process Operations” provides a central entry for all data consistency related tasks and procedures in the Data Consistency Management view, thus helping to analyse the root cause as well as to find relevant correction and recovery reports.

A guided procedure related approach allows creating a service session for detailed interaction, navigation and status tracking of a discovered data inconsistency. A status overview for each consistency analysis session is provided as well.
6 How to Measure the Success of the Implementation?

To assess the quality of the process, clearly-defined parameters and measurable objectives are required. The key parameters should be collated and evaluated in regular reports. The historical data that is created in this way can be used to identify trends and then derive the necessary measures. The following aspects could be considered:

- **Increased end user satisfaction in application support**
  - By reacting to possibly critical situations before the end users work and complete business process is impacted, the end user has to open fewer tickets and gains a higher trust in the solution and the work of the Application Support.
  - In order to measure this benefit, the number of tickets opened by the end users (either in total or in certain application areas) should be considered.
  - Measure the monthly number of tickets opened by end users in the relevant application areas. The monthly number of tickets opened before the implementation of a business process and interface monitoring concept should be compared to the monthly number of tickets opened after the implementation of the monitoring concept. Identify the percentage of tickets opened previously that would be avoided by implementing the monitoring concept. The target number of tickets should be the percentage that cannot be prevented by the monitoring concept.

- **Decreased business process downtime**
  - By reacting to possibly critical situations before they have become severe, the problem can be solved before the business process is brought to a standstill.
  - In order to measure this benefit, the total monthly business process downtime should be considered.
  - Measure the current monthly business process downtime. Identify which business process downtime could have been prevented by the monitoring concept. The target monthly business process downtime should be the time that could not have been prevented by the monitoring concept.
• Improved stability of business process
  o By monitoring the stability of the business processes, by solving alert situations in the business process execution, by reporting over the number of alerts and trends for the business process flow and by finding the root cause of critical trends and applying counter measures the business process stability improves and becomes stable at an improved level (considering that no changes to the solution have taken place)
  o In order to measure this benefit the number of alerts that have occurred should be considered.
  o Measure the number of alerts that have occurred. After a defined “warm up” period the number determine the number of alerts that can be handled by application support (both first and second level). The number of alerts should decline and stabilize at this level.

The different measures could include:
• Number of documented business processes
• Number of monitored processes
• Number of monitoring objects (interfaces, data consistency, business objects)
• Number of documented error handling procedures
• Time to fix inconsistency
• Availability of business processes
• Number of inconsistencies
• Number of alerts
• Number of tickets created by end users