



Session ID: SPC251 Unicode Interfaces – Data Exchange Between Unicode and non-Unicode Systems



Dr. Christian Hansen, SAP AG

Introduction

- About Code Pages
- Communication: The Ideal Picture
- Communication: The Reality

Part I – RFC SAP R3 ↔ SAP R3

- Unicode ↔ Unicode
- Unicode ↔ single code page system
- Unicode ↔ MDMP system

Part II – File transfer

- Writing and reading files on the application server
- Writing and reading files on the front end

Part III – Common mistakes

Exercises I

Part IV RFC SAP R3 ↔ RFC client

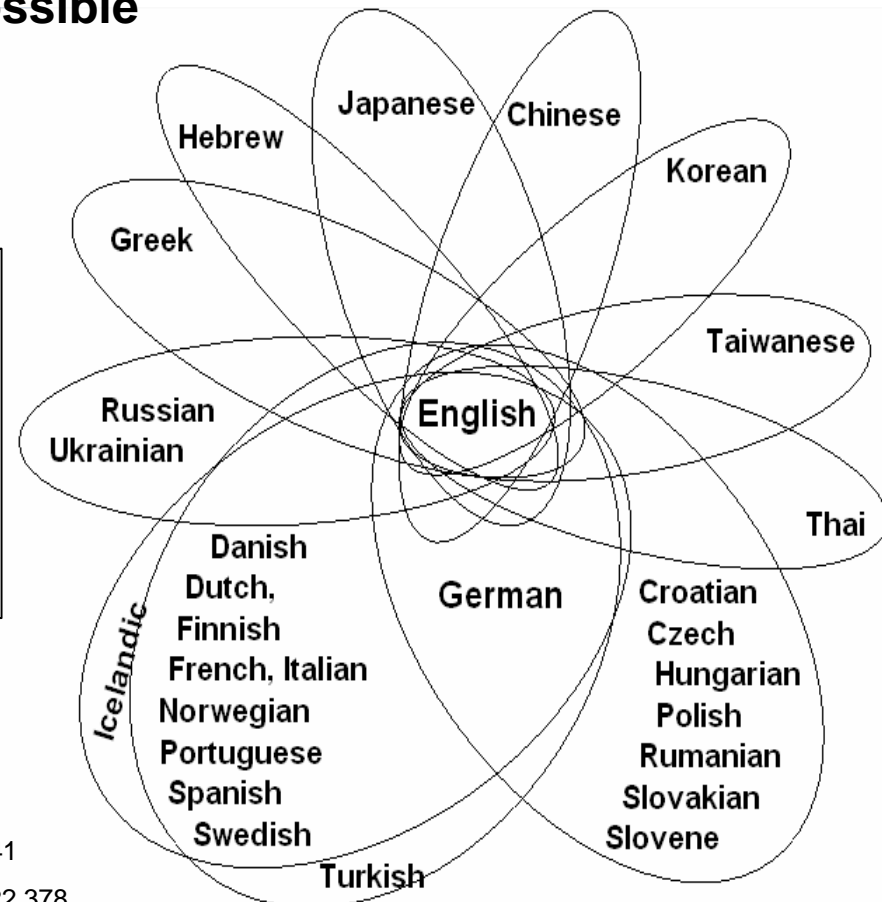
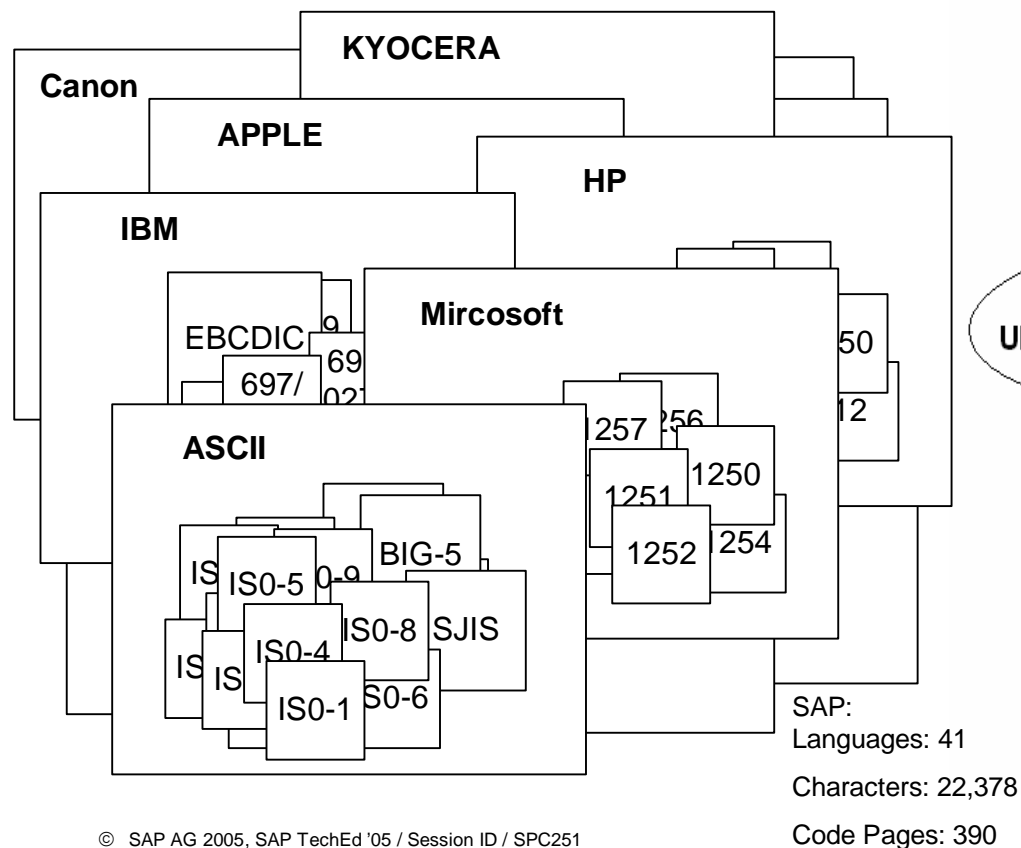
- Unicode interface for C programming
- Other languages



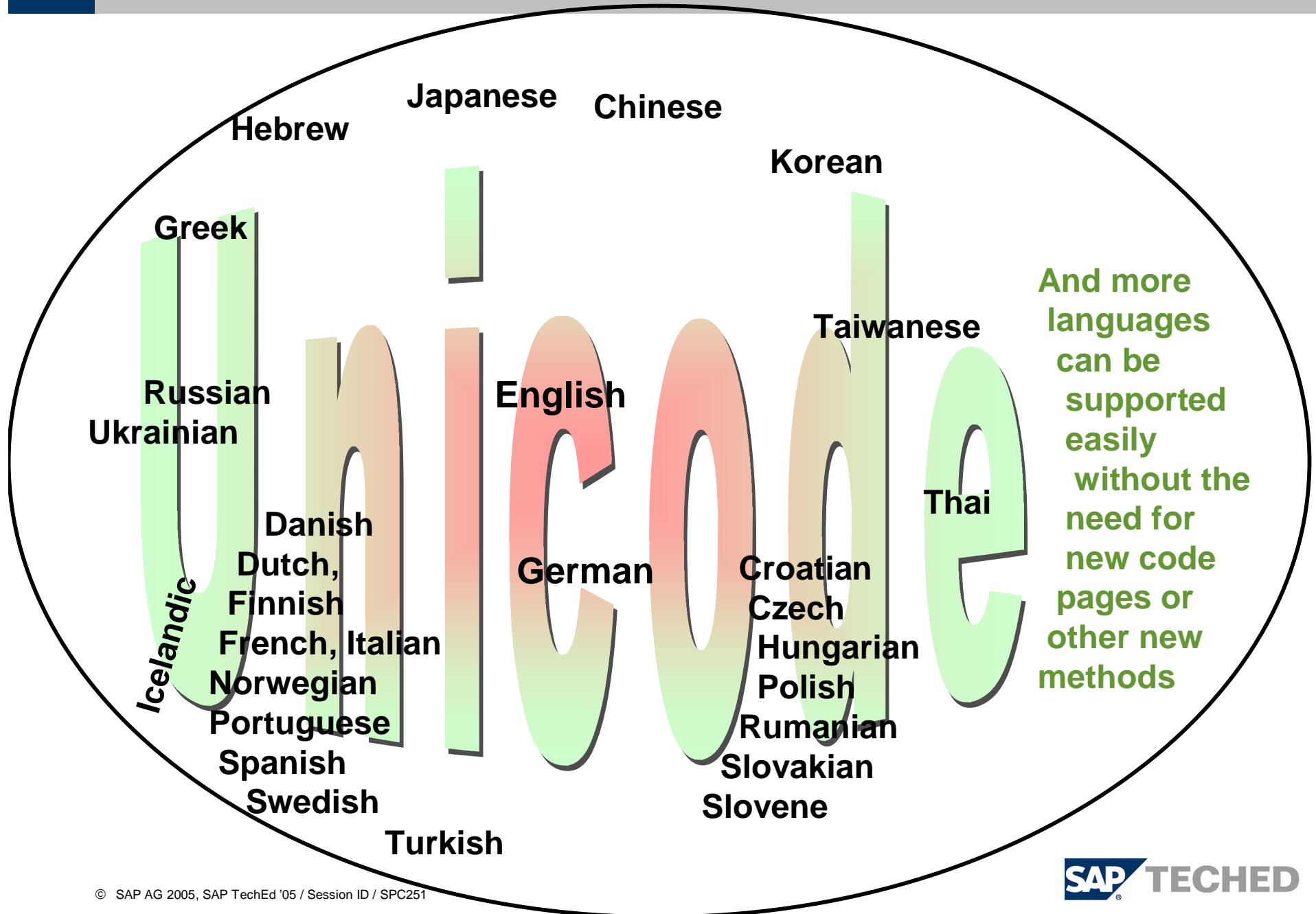
About Code Pages: Conventional Code Pages

Disadvantages of old standard code pages

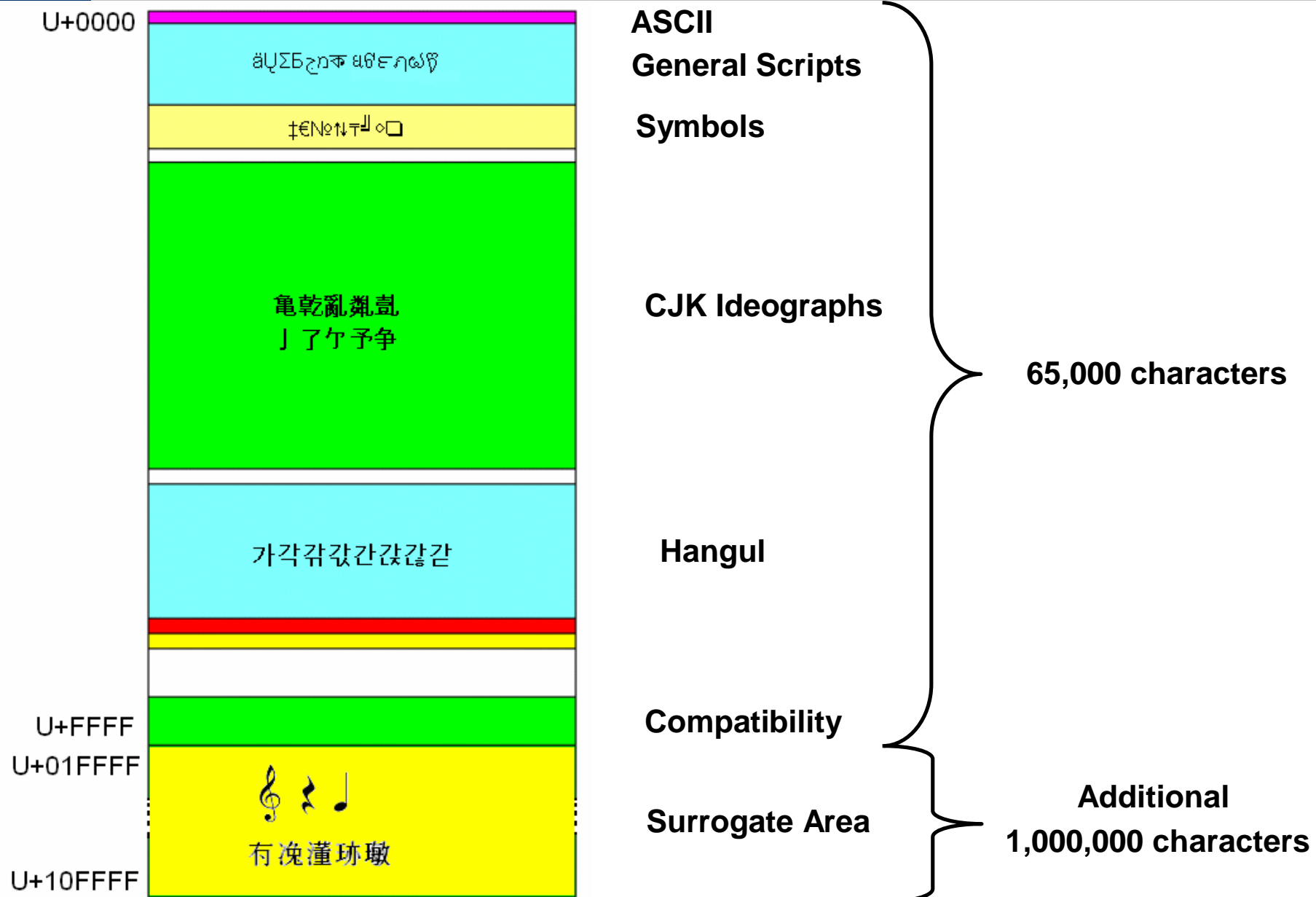
- Each covers only a subset of all characters used
- Incompatibilities between different codepages
- Only restricted data exchange possible
- Too many of them



Solution: Unicode, one Code Page for all Scripts



Solution: Unicode characters



Representation of Unicode Characters

UTF-16 – Unicode Transformation Format, 16 bit encoding

- Fixed length, 1 character = 2 bytes (surrogate pairs = 2 + 2 bytes)
- Platform-dependent byte order (big/little endian)
- 2 byte alignment restriction

UTF-8 – Unicode Transformation Format, 8 bit encoding

- Variable length, 1 character = 1...4 bytes
- Platform independent
- no alignment restriction
- 7 bit US ASCII compatible

Character	Unicode scalar value	UTF-16 big endian	UTF-16 little endian	UTF-8
a	U+0061	00 61	61 00	61
ä	U+00E4	00 E4	E4 00	C3 A4
α	U+03B1	03 B1	B1 03	CE B1
會	U+3479	34 79	79 34	E3 91 B9

Old solution for multiple languages: MDMP*

West European View

Table: Data Browser: Ta

COLOR	SPRAS	NAME
B	DE	blau
G	DE	grün
R	DE	rot
Y	DE	gelb
B	EN	blue
G	EN	green
R	EN	red
Y	EN	yellow
B	JA	青
G	JA	緑
R	JA	赤
Y	JA	黄
B	KO	파란색
G	KO	초록색
R	KO	빨간색
Y	KO	노란색

Japanese View

Table: データブラウザ

COLOR	SPRAS	NAME
B	DE	blau
G	DE	grün
R	DE	rot
Y	DE	gelb
B	EN	blue
G	EN	green
R	EN	red
Y	EN	yellow
B	JA	青
G	JA	緑
R	JA	赤
Y	JA	黄
B	KO	파란색
G	KO	초록색
R	KO	빨간색
Y	KO	노란색

Korean View

Table: 데이터브라우저

COLOR	SPRAS	NAME
B	DE	blau
G	DE	grün
R	DE	rot
Y	DE	gelb
B	EN	blue
G	EN	green
R	EN	red
Y	EN	yellow
B	JA	青
G	JA	緑
R	JA	赤
Y	JA	黄
B	KO	파란색
G	KO	초록색
R	KO	빨간색
Y	KO	노란색

* Check your system type with report RSCPINST → current configuration

Old solution for multiple languages: MDMP

West European View

COLOR	SPRAS	NAME
B	DE	blau
G		grün
R		rot
Y		gelb
B	EN	blue
G		green
R		red
Y		yellow
B	JA	青
G		緑
R		赤
Y		黄
B	KO	파란색
G		초록색
R		빨간색
Y		노란색

Japanese View

COLOR	SPRAS	NAME
B	DE	blau
G		grün
R		rot
Y		gelb
B	EN	blue
G		green
R		red
Y		yellow
B	JA	青
G		緑
R		赤
Y		黄
B	KO	파란색
G		초록색
R		빨간색
Y		노란색

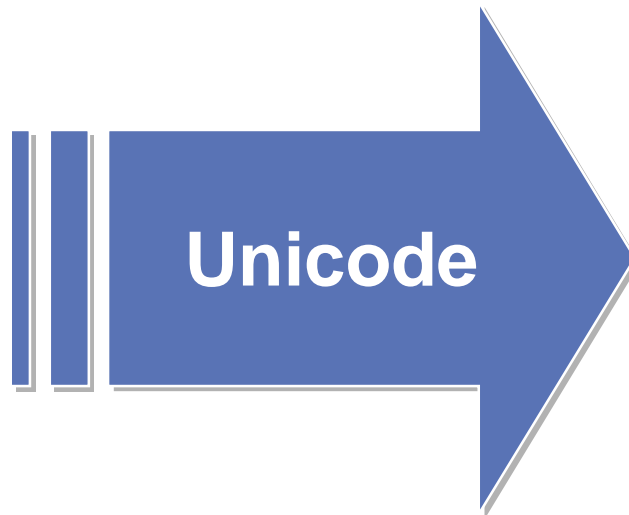
Korean View

COLOR	SPRAS	NAME
B	DE	blau
G		grün
R		rot
Y		gelb
B	EN	blue
G		green
R		red
Y		yellow
B	JA	파란색
G		초록색
R		빨간색
Y		노란색

End of support with NetWeaver 04
(see notes 838402 and 79991)

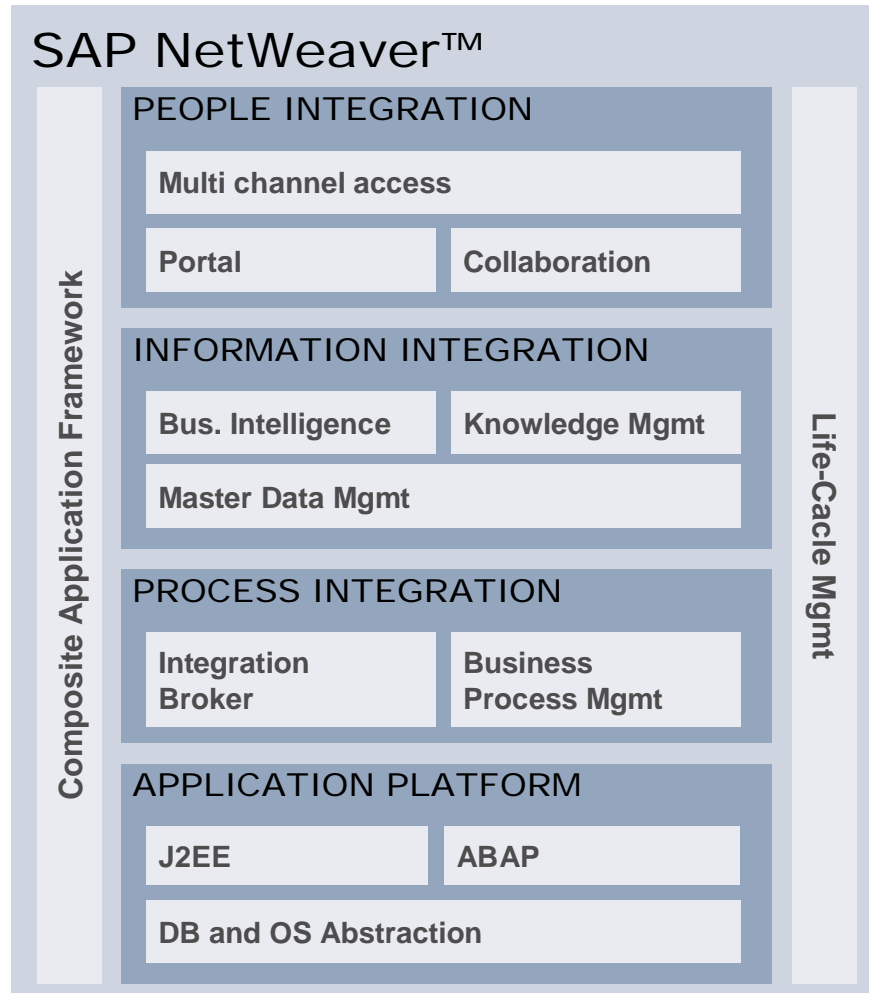
(As of release NetWeaver 04s and moving forward, MDMP will no longer be supported)

Only supported solution for multilingual systems: Unicode



COLOR	SPRAS	NAME
B	DE	blau
G	DE	grün
R	DE	rot
Y	DE	gelb
B	EN	blue
G	EN	green
R	EN	red
Y	EN	yellow
B	JA	青
G	JA	緑
R	JA	赤
Y	JA	黄
B	KO	파란색
G	KO	초록색
R	KO	빨간색
Y	KO	노란색

SAP NetWeaver™ the integration platform?



Evolution of mySAP Technology

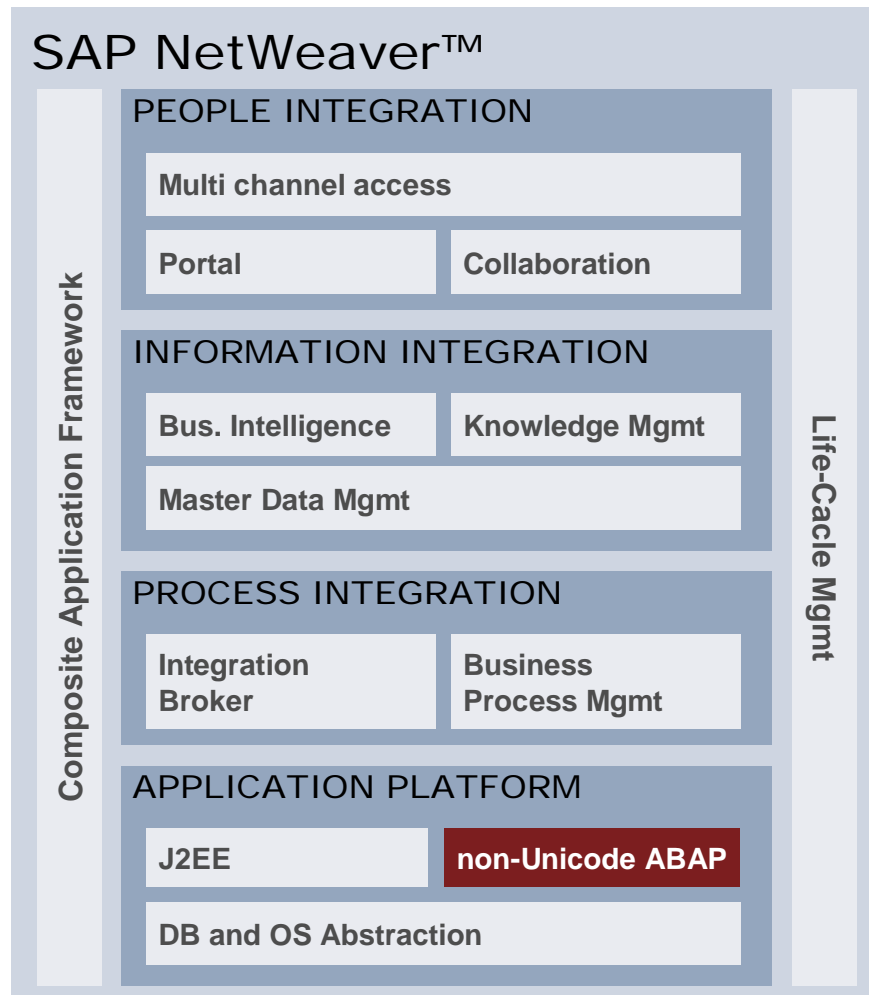
Unifies and aligns people, information and business processes

- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

SAP NetWeaver™ with non-Unicode ABAP stack



Evolution of mySAP Technology

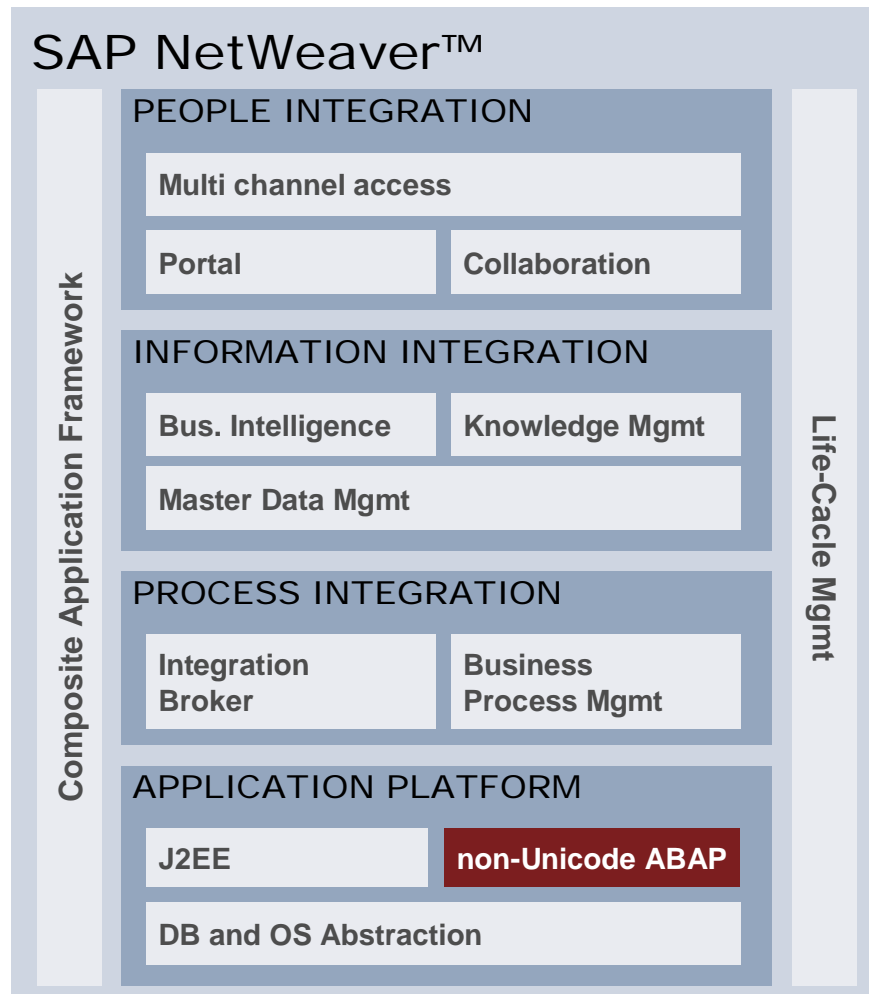
Unifies and aligns people, information and business processes

- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

SAP NetWeaver™ with non-Unicode ABAP stack



Evolution of mySAP Technology

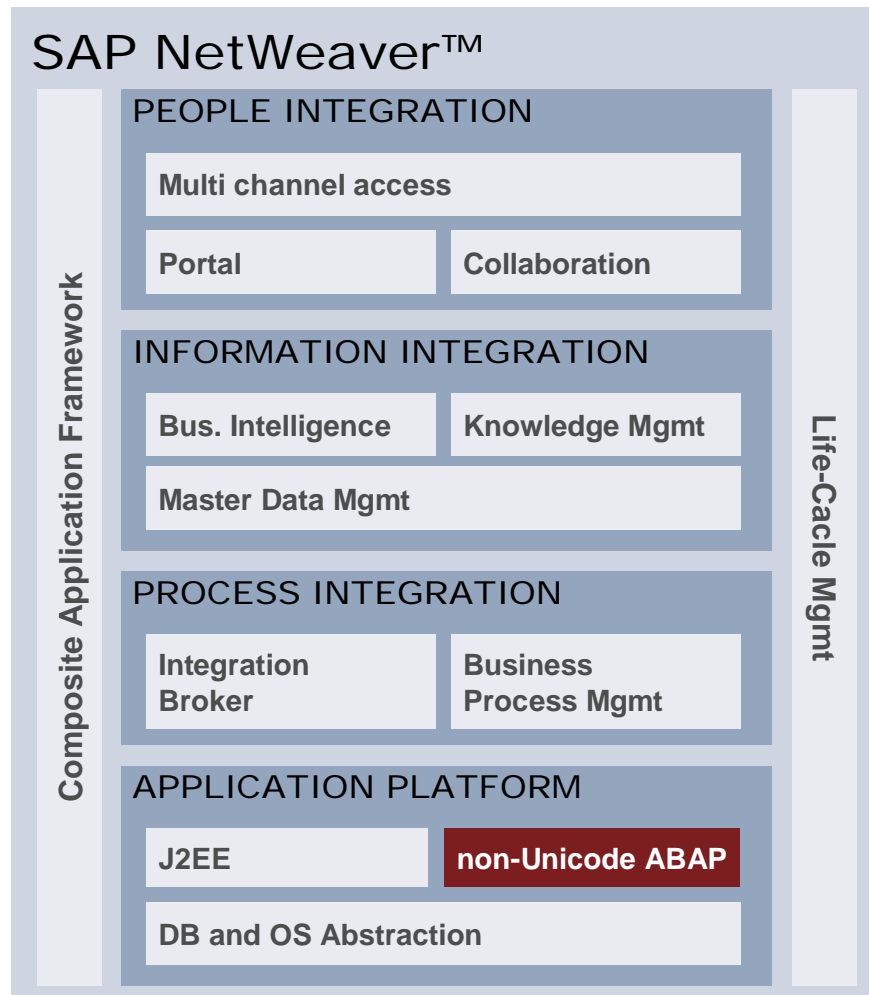
Unifies and aligns people, information and business processes

- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

SAP NetWeaver™ with non-Unicode ABAP stack



Evolution of mySAP Technology

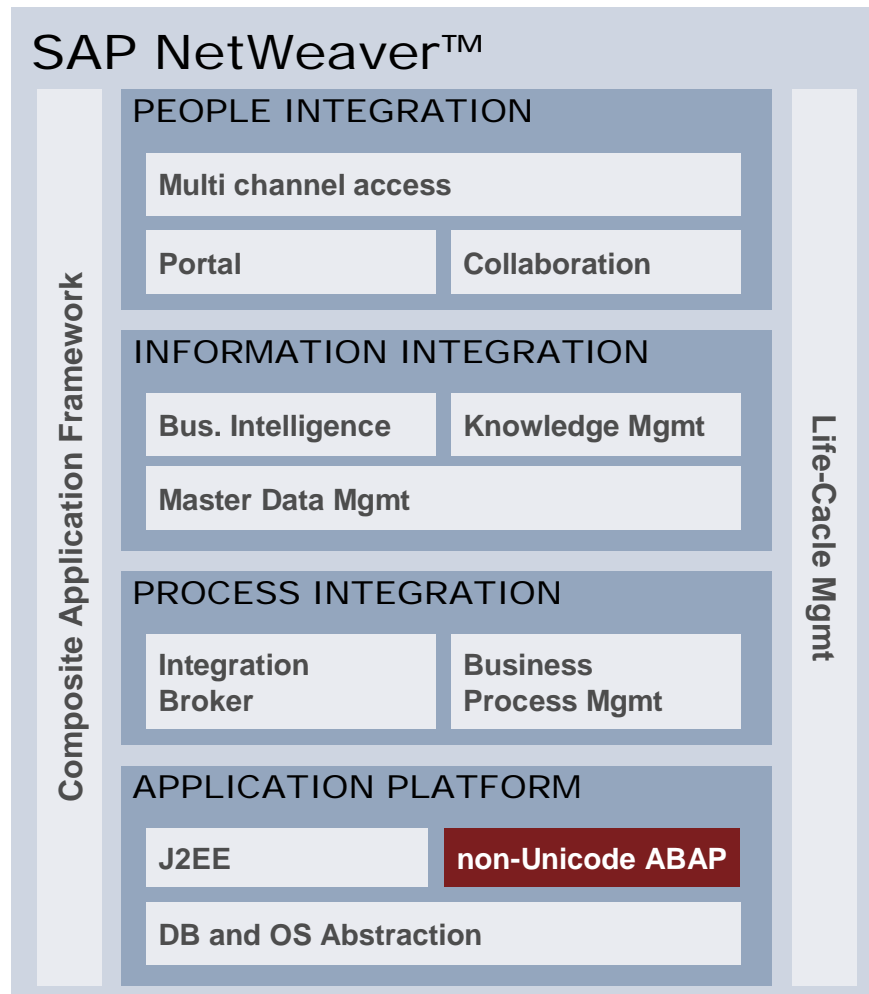
Unifies and aligns people, information and business processes

- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

SAP NetWeaver™ with non-Unicode ABAP stack



Evolution of mySAP Technology

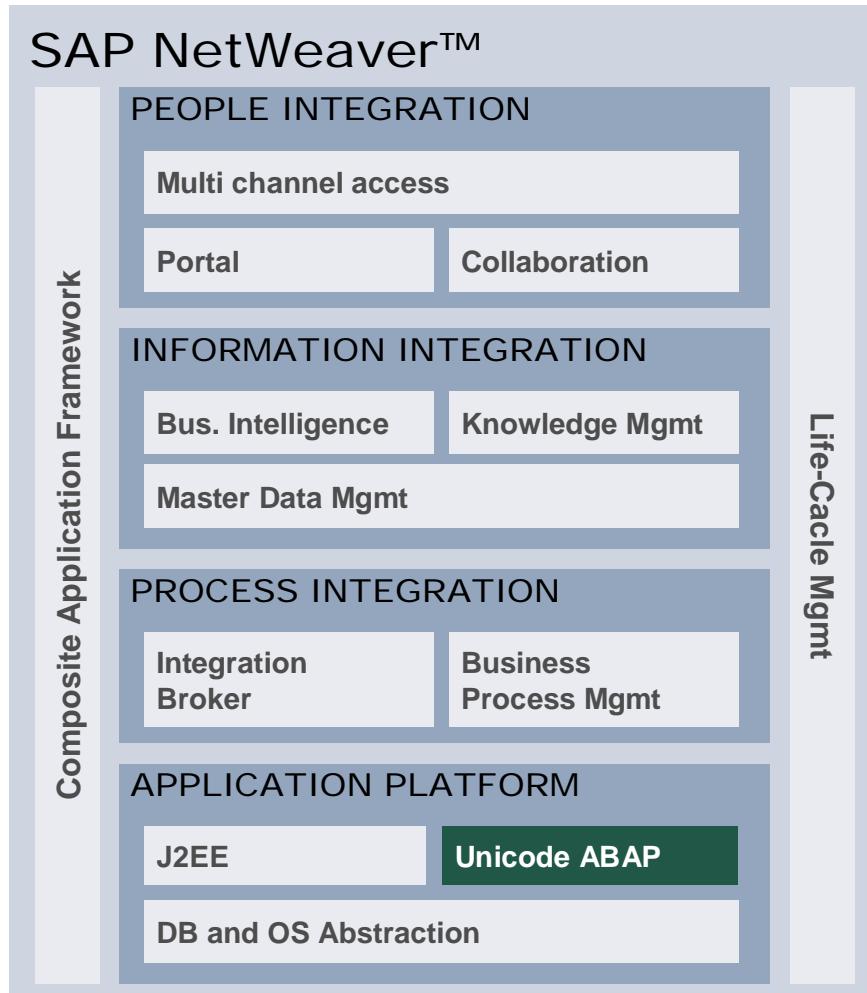
Unifies and aligns people, information and business processes

- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

Only solution for full integration: Unicode



Evolution of mySAP Technology

Unifies and aligns people, information and business processes

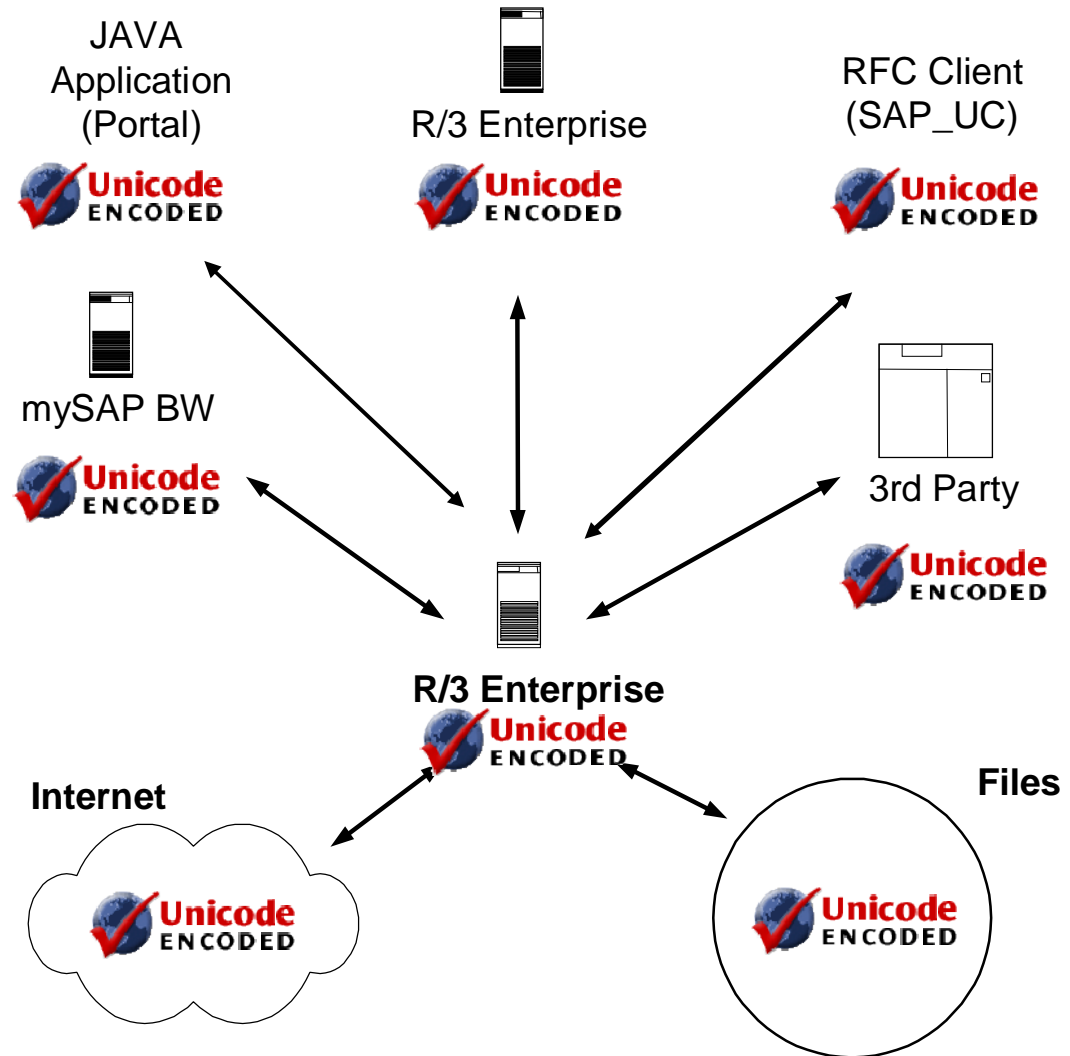
- Integrates across technologies and organizational boundaries
- A safe choice with full .NET and J2EE interoperability

The business foundation for SAP and partners

- Powers business-ready solutions that reduce custom integration
- Its Enterprise Services Architecture increases business process flexibility

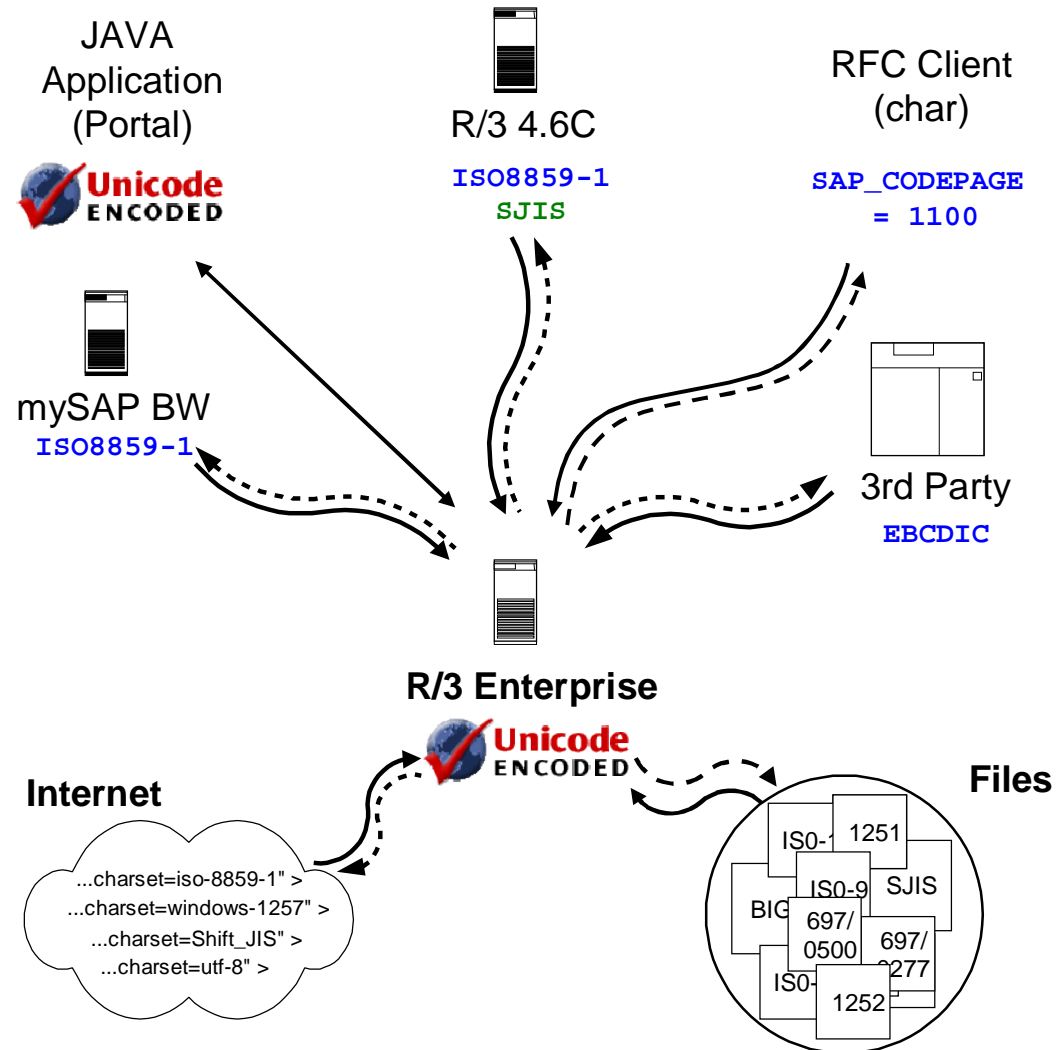
The ideal Picture: only Unicode components

- **Conversions are done algorithmically (1:1 relation)**
- **No data misinterpretation**
- **No data loss**
- **All business relevant characters available at the same time**
- ...



The reality: Unicode and non-Unicode components

- **Conversions between incompatible code pages everywhere**
- **Only common subset exchangeable**
- **Special rules have to be obeyed to make communication possible**
- ...



Introduction

- About Code Pages
- The Ideal Picture
- Reality

Part I – RFC

- Unicode ↔ Unicode
- Unicode ↔ single code page system
- Unicode ↔ MDMP system

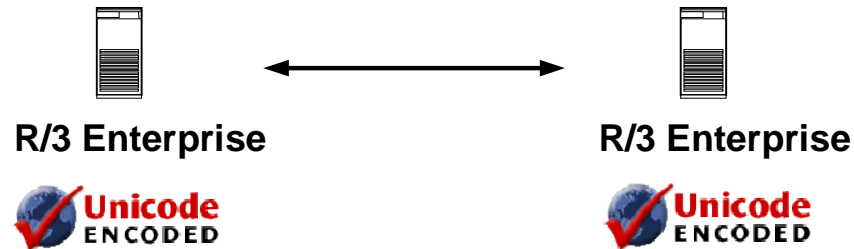
Part II – File transfer

- Writing and reading files on the application server
- Writing and reading files on the front end

Part III – Common mistakes

Exercises

RFC Unicode ↔ Unicode



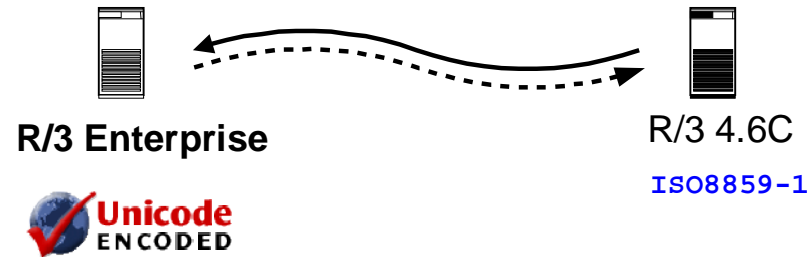
In case of an Unicode ↔ Unicode combination RFC passes all character data without code page conversion or merely with adaption of the endianness.

- UTF-16 big endian = SAP code page 4102
- UTF-16 little endian = SAP code page 4103

Information about the destination is maintained in SM59 →
→ special options → character width in target system

- 1 Byte = non-Unicode
- 2 Byte = Unicode

RFC Unicode ↔ non-Unicode single code page

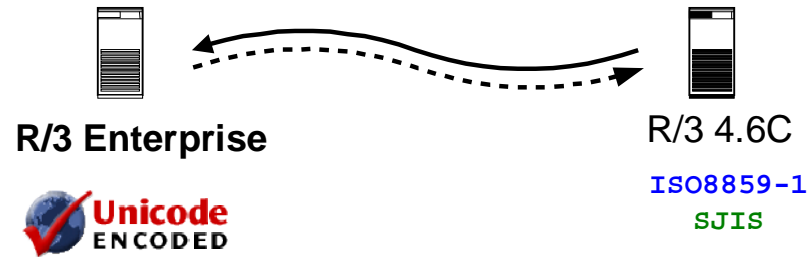


In case of an Unicode ↔ non-Unicode single code page combination, RFC passes all character data with code page conversion between Unicode and the old code page.

As Unicode is a true superset of any old standard codepage not all Unicode characters can be transferred to the non-Unicode system:

Ä	↔	Ä
ß	↔	ß
あ	→	#
東	→	#
한	→	#
₩	→	#

RFC Unicode ↔ non-Unicode MDMP



In case of an Unicode ↔ non-Unicode MDMP combination RFC passes all character data with code page conversion between Unicode and the different old code pages.

Which of the MDMP code pages is chosen depends on the language:

Ä	← DE →	Ä
ß	← DE →	ß
あ	← JA →	あ
東	← JA →	東
한	→	#
₩	→	#

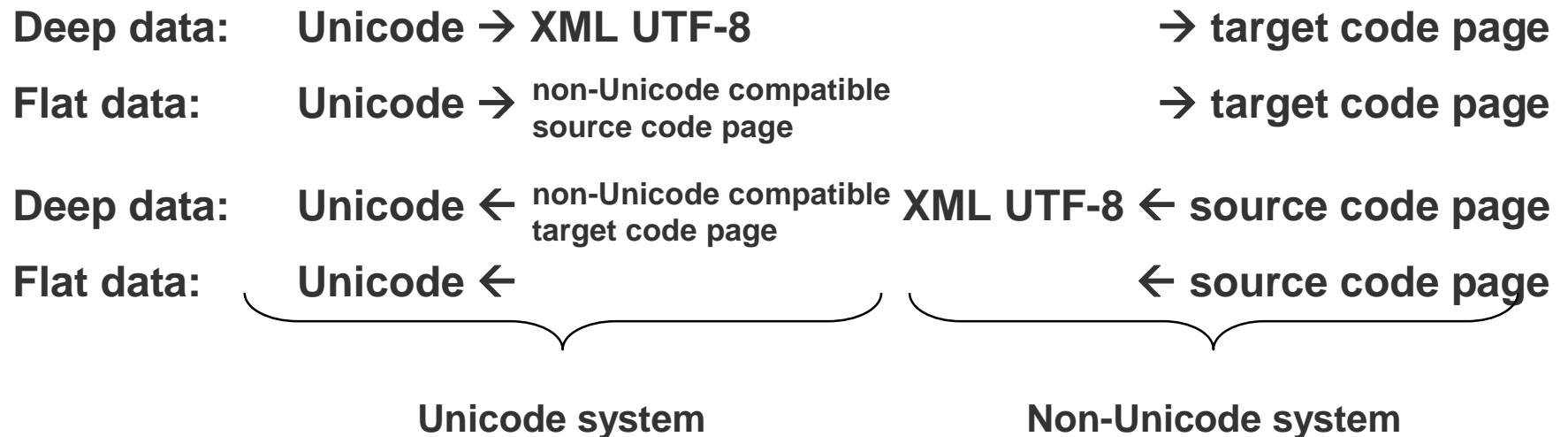
Excursion: Difference between “flat” and “deep” data types

- **Flat:** C, N, D, T, X, I, F, P and any structure consisting only of these fields
- **Deep:** STRING, XSTRING, table types, object references and any structure containing one of these types

Deep data types are transferred using an UTF-8 encoded XML format (XRFC).

Excursion: Difference between “flat” and “deep” data types

Detailed conversion paths:



Deriving code pages a): Data without text language field

Source system	Data type	Source code page	Intermediate format *	Target code page
Unicode	Flat	Unicode	Logon language source system * *	Logon language target system
	Deep		UTF-8 based XML	
non-Unicode	Flat	Logon language source system	Logon language source system	Unicode
	Deep	SY-LANGU source system	UTF-8 based XML	

* XML / non-Unicode compatible code page

* * You may switch to "Logon language target system" using RFC bit option 0x200 at SM59 → Special options → RFC Bit Options

Example: Flat data, logon language German

Ä	← Logon = DE →	Ä
あ	Logon = DE →	#

Deriving code pages b) : Data (flat) with text language field

Flat Structures containing a language field (domain SPRAS, DDIC data type LANG) with maintained text language flag have a special handling:

For TABLES parameters automatic language code page assignment is done during RFC for each row independent of logon language.

This enables sending and receiving tables from MDMP systems (different code pages for each row):

Ä
あ

← Logon = DE / text language = DE →
← Logon = DE / text language = JA →

Ä
あ

Deriving code pages b) : Data (flat) with text language field

Automatic code page assignment is activated/configured by the following actions in the Unicode system:

- **Maintaining text language flag with SE11**
- **Maintaining language code page assignment with SM59 for outgoing calls**
- **Maintaining language code page assignment for incoming calls from legacy systems that don't pass their code page information with a pro forma destination (note 722193)**

Maintain RFC destination SM59: MDMP settings

The screenshot shows the SAP SM59 interface for maintaining an RFC destination named 'B20_hansenc_mdmp'. The 'Technische Einstellungen' (Technical Settings) tab is active, showing the 'MDMP-Einstellung' (MDMP Setting) section. The 'Aktiv' (Active) radio button is selected and circled in green. A green arrow points from this button to a floating window titled 'RFC Destination B20_hansenc_mdmp' which displays the 'MDMP Codepage Mapping' table.

MDMP Codepage Mapping Table:

Lang.	Local	Remote
KO	8500	8500
DE	1100	1100
EN	1100	1100
JA	8000	8000

SE11: Maintain text language flag

The screenshot shows the SAP SE11 'Display Field' dialog for MAKT-SPRAS. The main window displays the 'Fields' tab with a table of fields. The 'SPRAS' field is highlighted with a green oval. A green arrow points from this field to the 'Text Lang.' checkbox in the 'Database Attributes' section of the dialog, which is also highlighted with a green oval.

Field	Key	Initi...	Data element	Data Ty...	L
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	
MATNR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MATNR	CHAR	
SPRAS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SPRAS	LANG	
MAKTX	<input type="checkbox"/>	<input type="checkbox"/>	MAKTX	CHAR	
MAKTG	<input type="checkbox"/>	<input type="checkbox"/>	MAKTG	CHAR	

Display Field MAKT-SPRAS

Data element attribute

Data element	SPRAS
Short Description	Language key
Domain	SPRAS
Data Type	LANG
No. of Characters	1
Default field name	LANGU

Possible entries

Origin of the input help	Input help
Search help name	H_T002
<input type="checkbox"/> Fixed val. ex.	Domain
<input checked="" type="checkbox"/> Foreign key exists	Check tab

Database Attributes

<input checked="" type="checkbox"/> Key field	<input checked="" type="checkbox"/> Initial
<input checked="" type="checkbox"/> Text Lang.	

Deriving code pages b) : Data (flat) with text language field

Frequent problem: data with invalid or unknown text language:

Ä text language = DE →

あ text language = JA →

Ö text language = éÂ →

ß text language = ES →

Ä

あ

runtime error RFC_CONVERSION_TABLE

runtime error RFC_CONVERSION_TABLE

If it is not possible to delete the wrong or obsolete data in the source system proceed as described in the note 809279 “RFC non-Unicode to Unicode with unknown text language”

Introduction

- About Code Pages
- The Ideal Picture
- Reality

Part I – RFC

- Unicode ↔ Unicode
- Unicode ↔ single code page system
- Unicode ↔ MDMP system

Part II – File transfer

- Writing and reading files on the application server
- Writing and reading files on the front end

Part III – Common mistakes

Exercises

Pattern for writing/reading files on the application server:

**OPEN DATASET IN <modus> MODE
TRANSFER/READ
CLOSE DATASET**

<modus>:

- **BINARY MODE**
Uninterpreted sequence of bytes.
- **TEXT MODE ENCODING UTF-8 / NON-UNICODE / DEFAULT**
Pure unstructured text data. DEFAULT equals UTF-8 in Unicode systems and NON-UNICODE in non-Unicode systems.
- **LEGACY TEXT/BINARY MODE**
Produces an format compatible to non-Unicode systems. Text data is always written in NON-UNICODE format. Not character-like structures are allowed. The only difference between TEXT and BINARY is, that in case of TEXT an EOF (END OF FILE) marker is added.

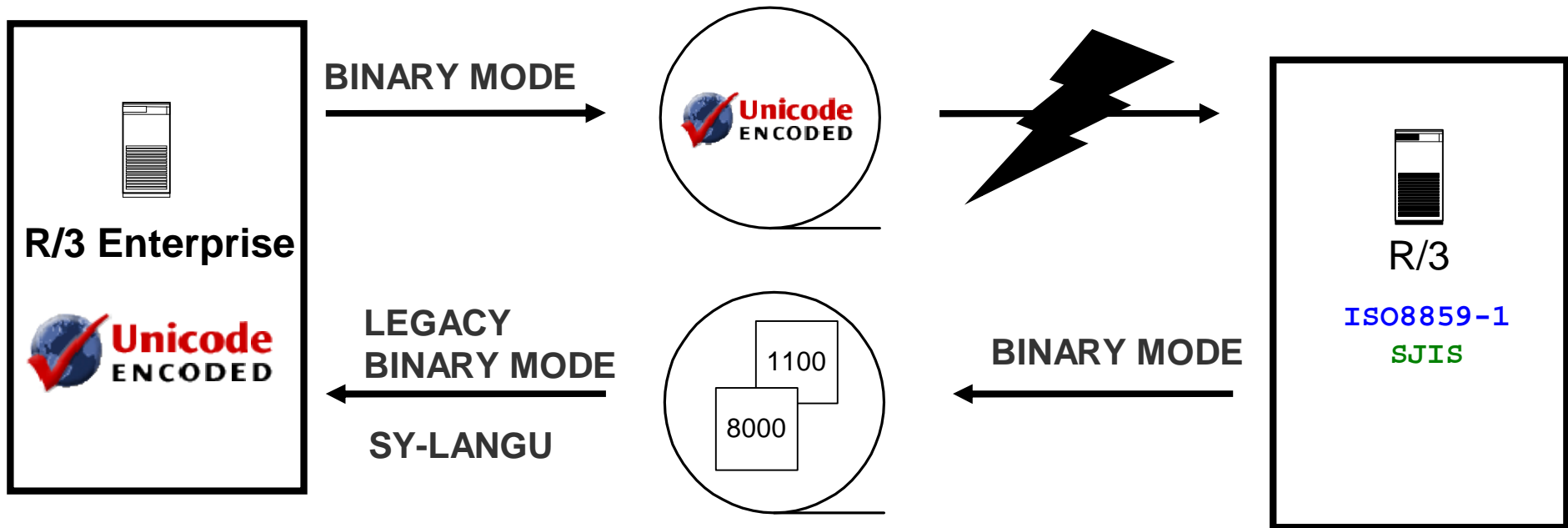
Code page selection NON-UNICODE:

If during data transfer a Unicode \leftrightarrow non-Unicode conversion is necessary, the non-Unicode code page is derived from the current system language SY-LANGU, which may be changed by using SET LOCALE LANGUAGE <lang>.

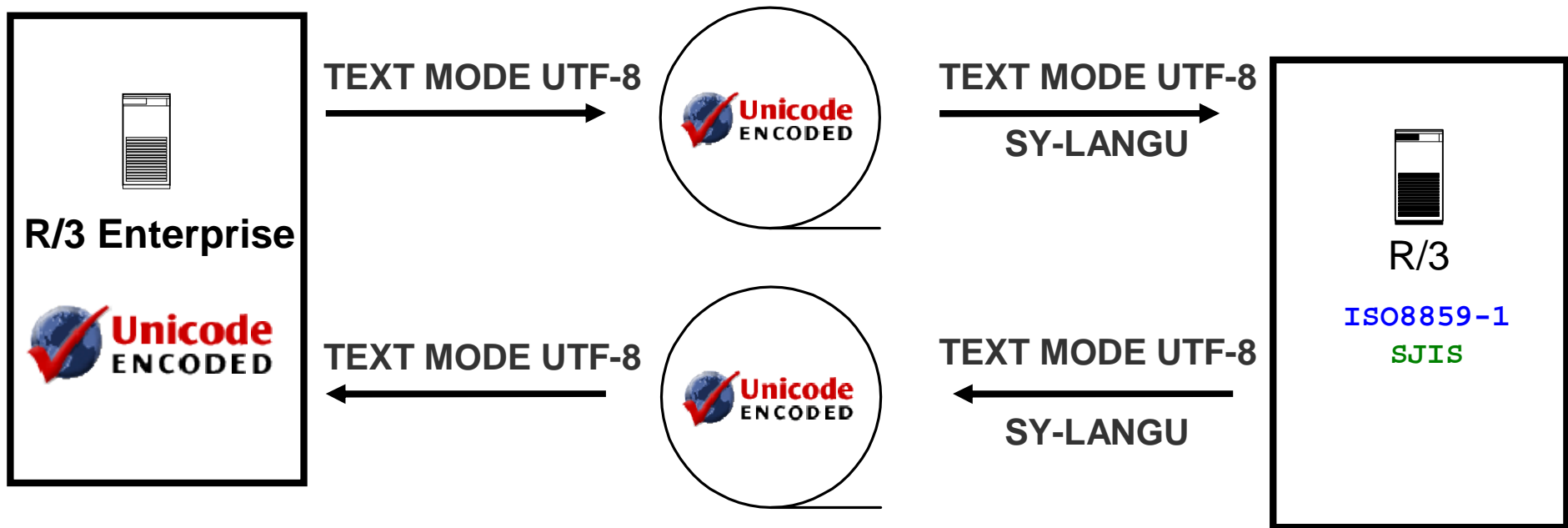
Advantages and disadvantages for data exchange:

- **BINARY.** Not a good exchange format in itself. Use this for writing/reading prepared data of well known format (e.g. XML /UTF-8 as XSTRING) or use for write/read on the same application server
- **TEXT MODE:** UTF-8 is a good exchange format. Structures may not be transferred as a whole. Only single fields
- **LEGACY MODES:** Only for reading or writing non-Unicode data. Structure and code page information is considered.

Example 1: BINARY MODE



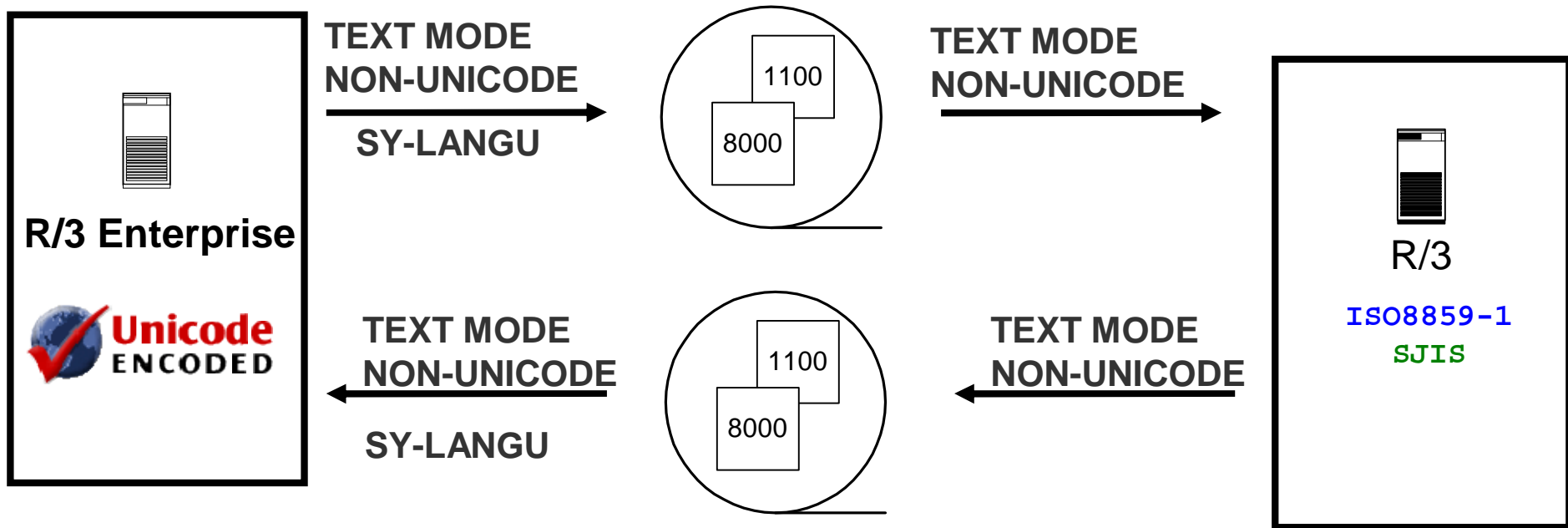
Example 2: TEXT MODE UTF-8



😊 Full charset supported (no data loss in the file)

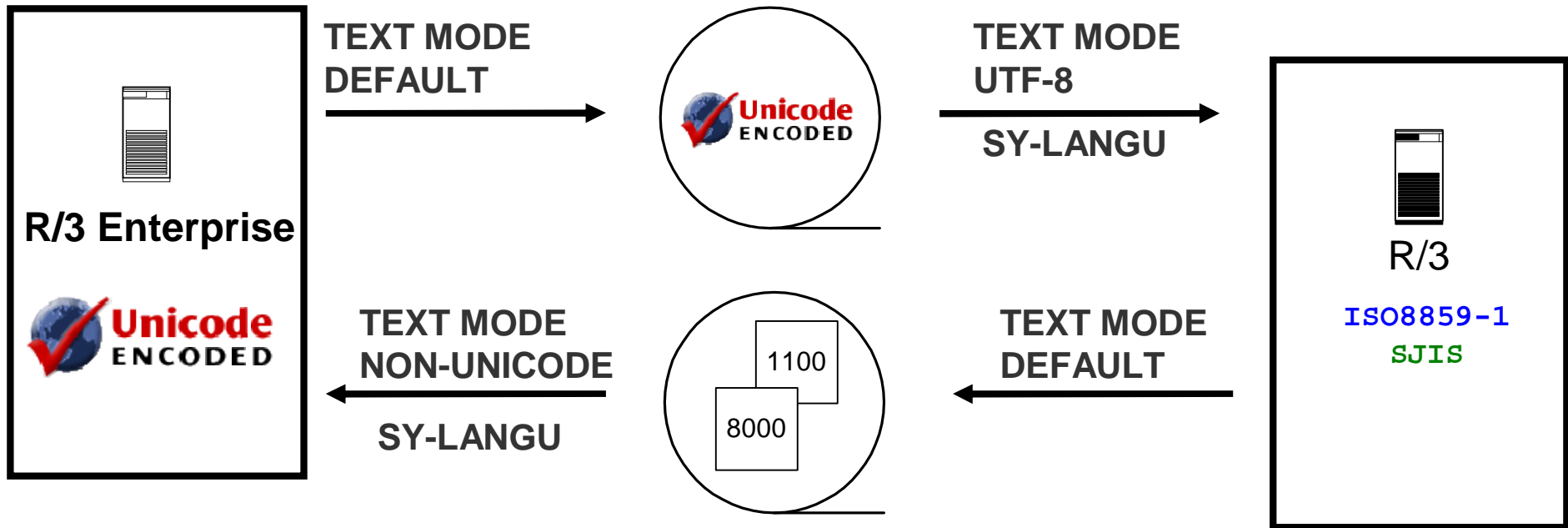
😞 Structured data as a whole → write field by field = 😊

Example 3: TEXT MODE NON-UNICODE

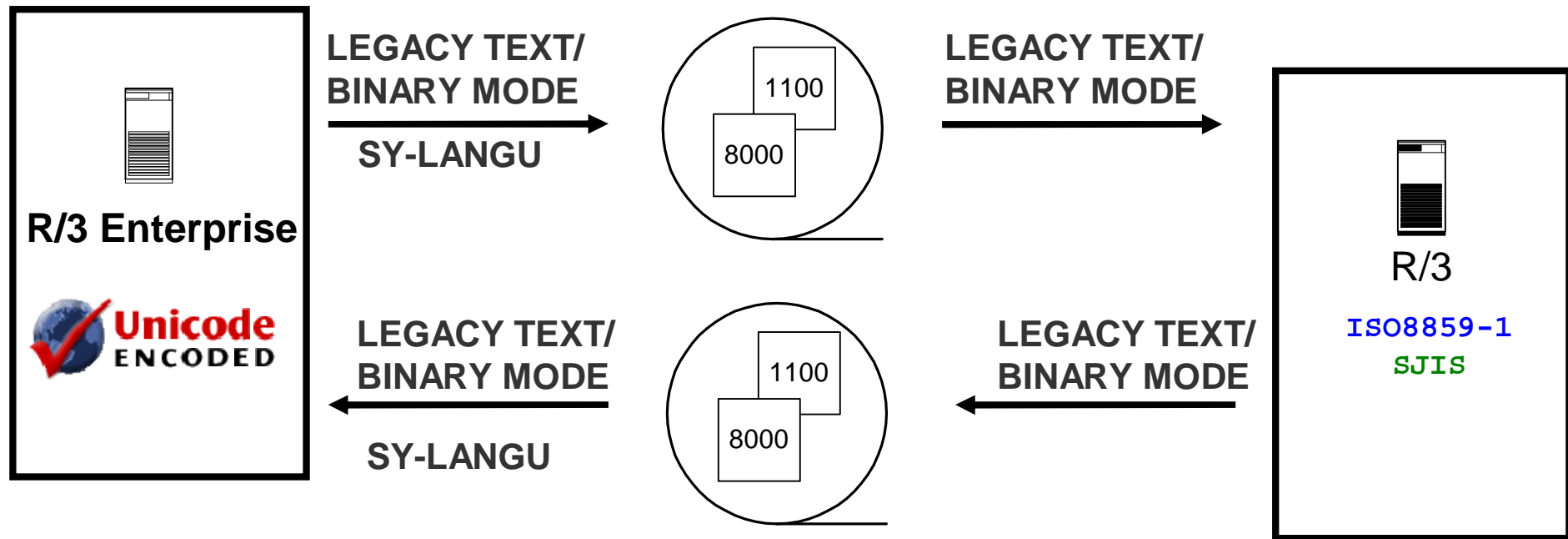


- ☹ Only part of UC charset supported (possible data loss in the file)
- ☹ Structured data as a whole → write field by field = 😊

Example 4: TEXT MODE DEFAULT



Example 5: LEGACY TEXT/BINARY MODE



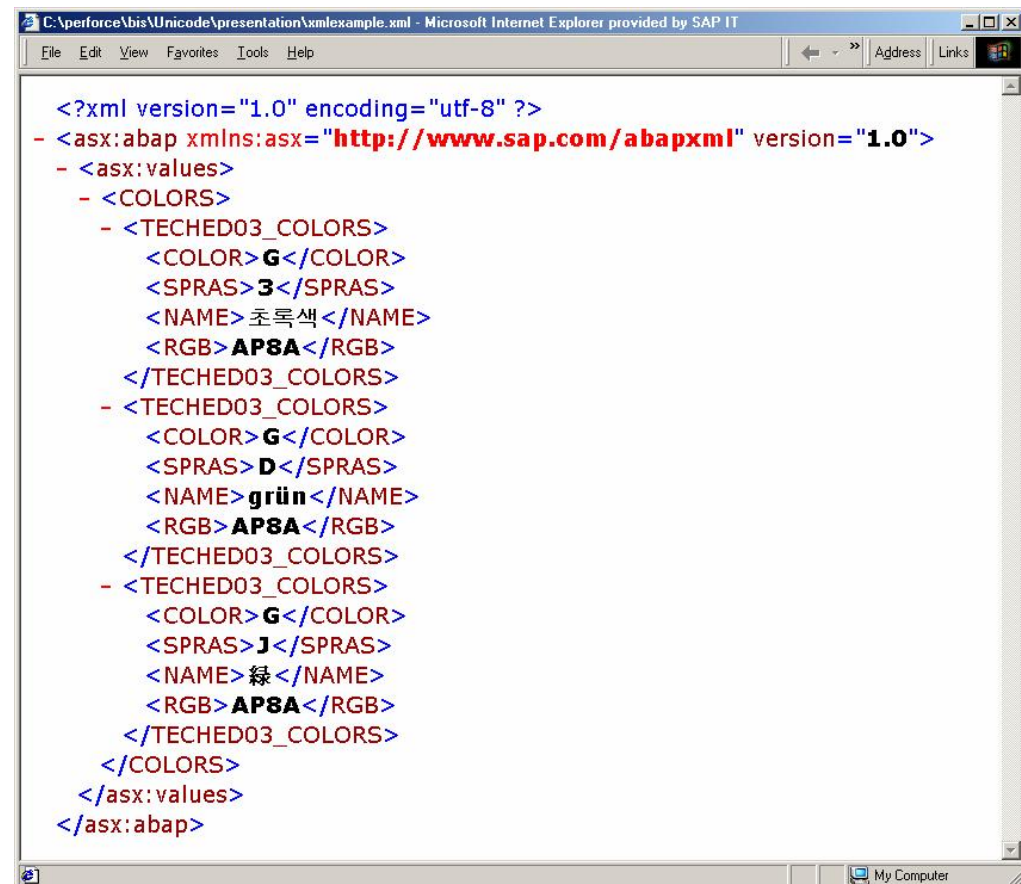
☹️ Only part of UC charset supported (possible data loss in the file)

😊 Structured data

Using XML as transport format

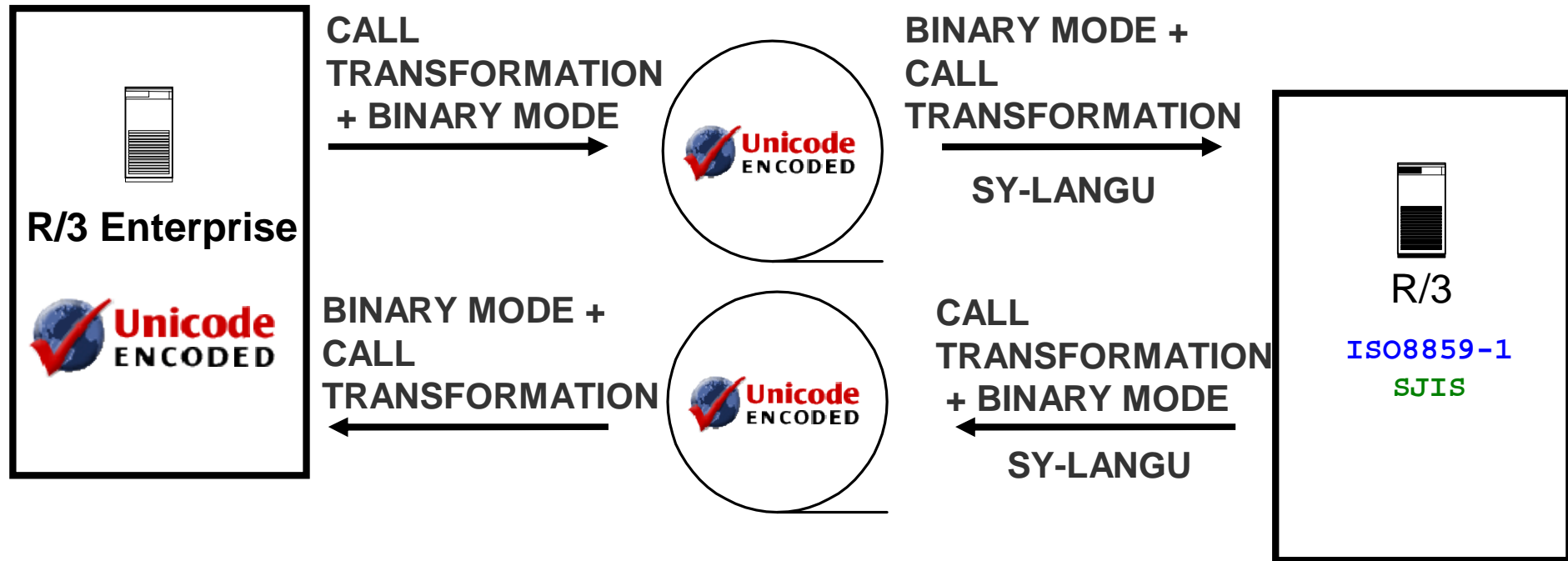
Use **CALL TRANSFORMATION** with target data type **XSTRING** to create an UTF-8 based XML representation of your data.

- Structure information (no layout / alignment problems)
- UTF-8 based (no data loss)
- Transport in binary form



```
<?xml version="1.0" encoding="utf-8" ?>
- <asx:abap xmlns:asx="http://www.sap.com/abapxml" version="1.0">
- <asx:values>
- <COLORS>
- <TECHED03_COLORS>
  <COLOR>G</COLOR>
  <SPRAS>3</SPRAS>
  <NAME>초록색</NAME>
  <RGB>APSA</RGB>
</TECHED03_COLORS>
- <TECHED03_COLORS>
  <COLOR>G</COLOR>
  <SPRAS>D</SPRAS>
  <NAME>grün</NAME>
  <RGB>APSA</RGB>
</TECHED03_COLORS>
- <TECHED03_COLORS>
  <COLOR>G</COLOR>
  <SPRAS>J</SPRAS>
  <NAME>緑</NAME>
  <RGB>APSA</RGB>
</TECHED03_COLORS>
</COLORS>
</asx:values>
</asx:abap>
```


Example 6: UTF-8 based XML + BINARY MODE



- ☺ Full charset supported (no data loss in the file)
- ☺ Structured data

Additional Remarks I:

- **The only Unicode Format supported with OPEN DATASET is UTF-8. UTF-16 is not supported**
- **If you still need to read Unicode and non-Unicode files at the same time, you have to provide users the chance to select the outside code page (see note 752835 Usage of the file interfaces in Unicode systems). OPEN DATASET ENCODING DEFAULT does not do it for you .**

As workaround for missing selection possibilities you may want to convert the files on the file system prior to loading them. For this purpose you may use the following tools:

- ◆ **sapiconv** **note 752859**
- ◆ **RSCP_CONVERT_FILE** **note 747615**

Additional Remarks II: Byte Order Mark (BOM)

Byte-Order-Marks (BOM, U+FEFF) are used to indicate the encoding of Unicode files and are not part of the file content.

SAPs recommendation:

- **Automatically detect UTF-8 encoded files with:
CL_ABAP_FILE_UTILITIES=>CHECK_FOR_BOM**
- **Read UTF-8 files with:
OPEN DATASET IN TEXT MODE ENCODING UTF-8 FOR INPUT
SKIPPING BYTE ORDER MARK**
- **Write files always with BOM:
OPEN DATASET IN TEXT MODE ENCODING UTF-8 FOR OUTPUT WITH
BYTE ORDER MARK**

See also SDN Weblog <https://weblogs.sdn.sap.com/pub/wlg/2194>

File transfer at the frontend with GUI_UP/DOWNLOAD

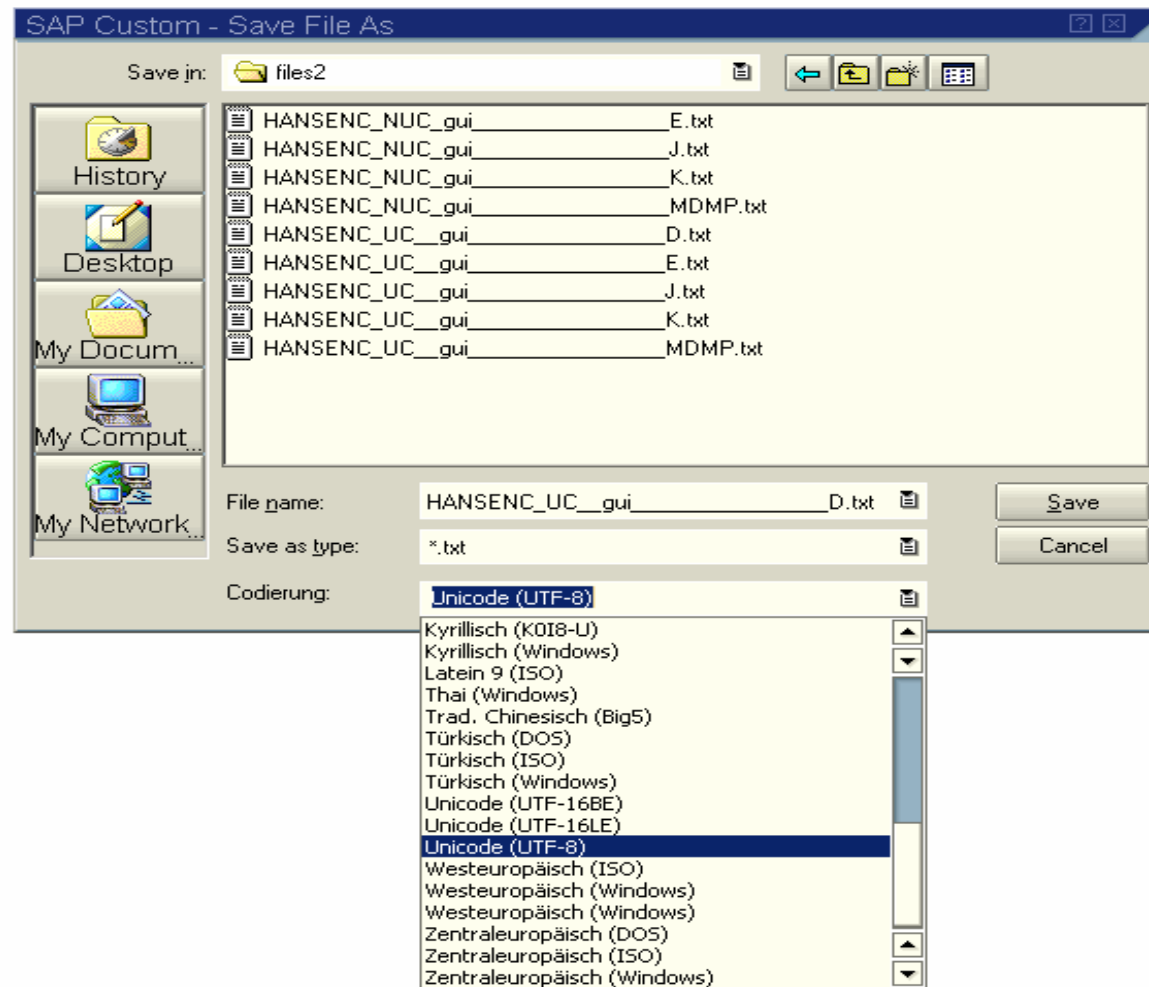
The function modules **GUI_/UPDOWNLOAD** convert data into textual representation. Structures are allowed.

Determination of the outside code page:

- Front end code page matching to the current system code page (SY-LANGU, SET LOCALE LANGUAGE)
- Declared explicitly with optional parameter CODEPAGE (Starting with release 6.20 SP 21).
- In interactive mode use
CL_GUI_FRONTEND_SERVICES=>FILE_OPEN/SAVE_DIALOG
(parameter WITH_ENCODING = 'X') to select the code page.
(only SAPGUI 6.40)

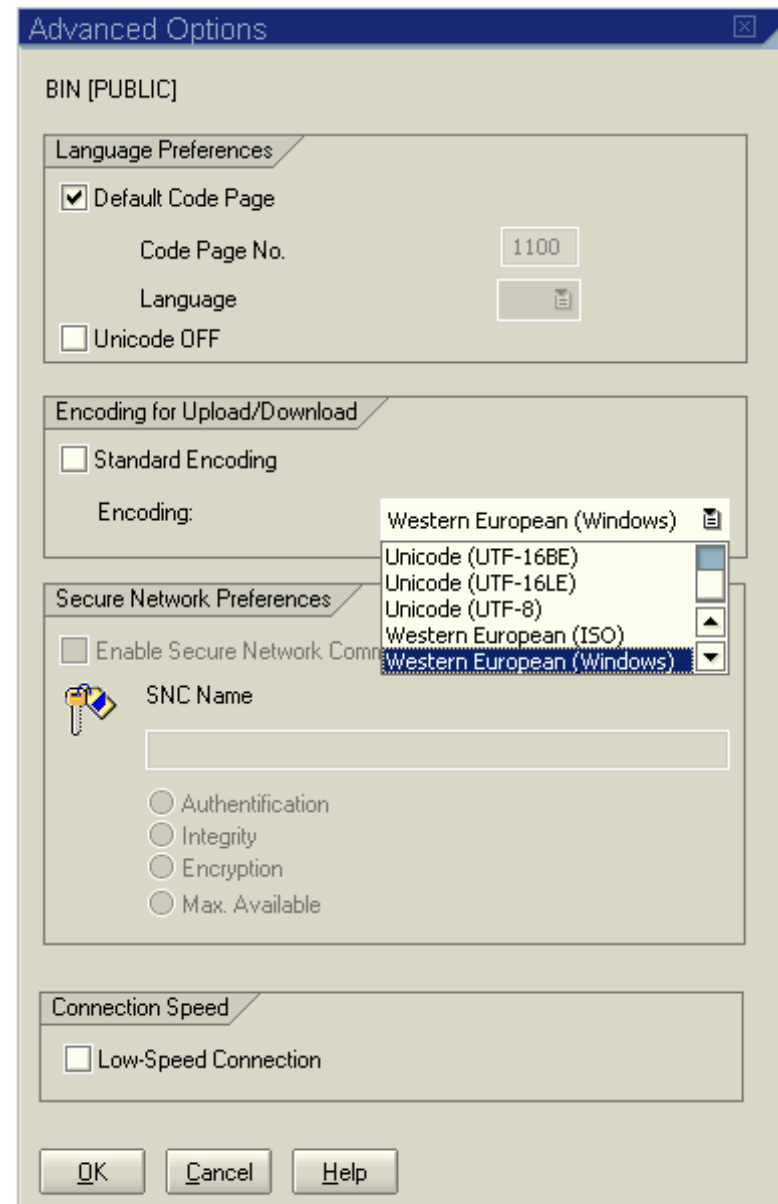
File transfer: Frontend

cl_gui_frontend_services=>file_open/save_dialog



Change default Encoding for Upload/Download :

SAP Logon → Properties → Advanced



RFC and file transfer from a Unicode systems perspective

			Target code page derived from	Suited for structured data	Reading/Writing single records	Transfer of "MDMP data" possible
XML			SY-LANGU	Yes	No	No
OPEN DATASET	TEXT MODE		SY-LANGU	Yes (field by field)	Yes	Yes
	LEGACY	TEXT MODE	SY-LANGU	Yes	Yes	Yes
		BINARY MODE	SY-LANGU	Yes	Yes	Yes
GUI	UPLOAD		SY-LANGU	Yes	No	No
	DOWNLOAD		SY-LANGU	Yes	Yes (Append)	Yes
RFC	flat	With language key	Value of language key	Yes	-	Yes
		Without language key	Logon language	Yes	-	No
	deep		Logon language	Yes	-	No

Introduction

- About Code Pages
- The Ideal Picture
- Reality

Part I – RFC

- Unicode ↔ Unicode
- Unicode ↔ single code page system
- Unicode ↔ MDMP system

Part II – File transfer

- Writing and reading files on the application server
- Writing and reading files on the front end

Part III – Common mistakes

Exercises

Things you should never do!



Type hiding



Missing text language field



Wrong length assumptions



Sending data that is not in the receivers codepage



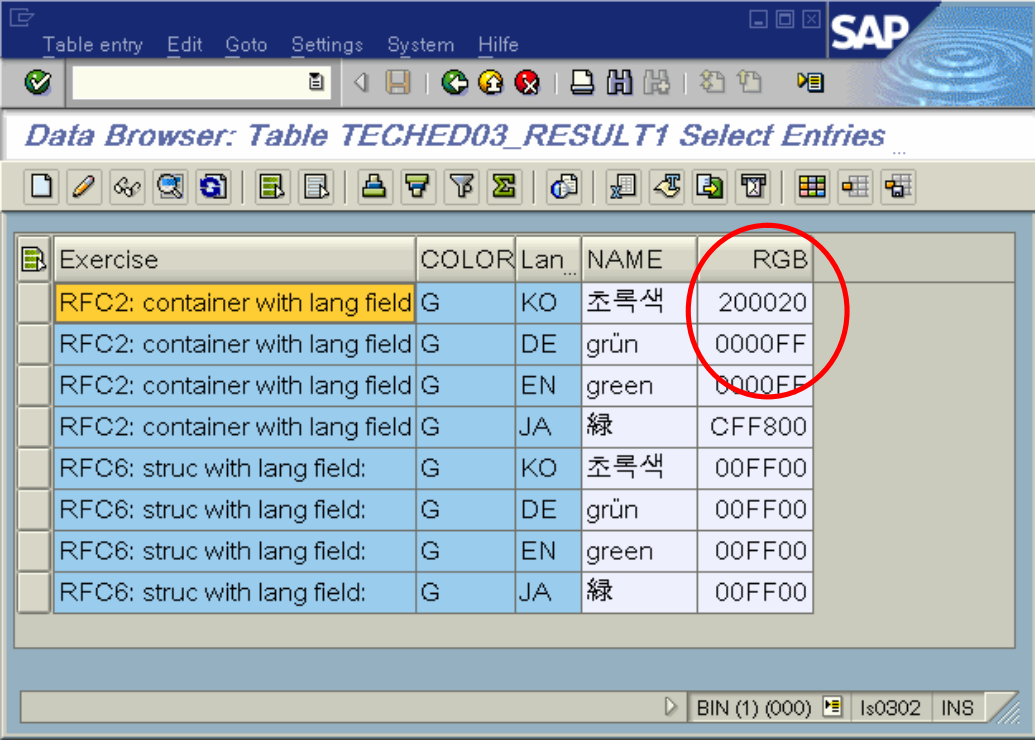
...

Common mistakes: Type hiding: binary data

Don't hide types 1

If you conceal the true types from the system the system cannot anything for you. As a consequence, data may, for example, be subject to unwanted codepage conversions.

Example: ☠ Transporting binary data in character containers



The screenshot shows the SAP Data Browser interface for table `TECHED03_RESULT1`. The table has columns: Exercise, COLOR, Lan..., NAME, and RGB. The RGB column contains hexadecimal values. A red circle highlights the first row's RGB value, 200020, which is a binary value stored in a character field.

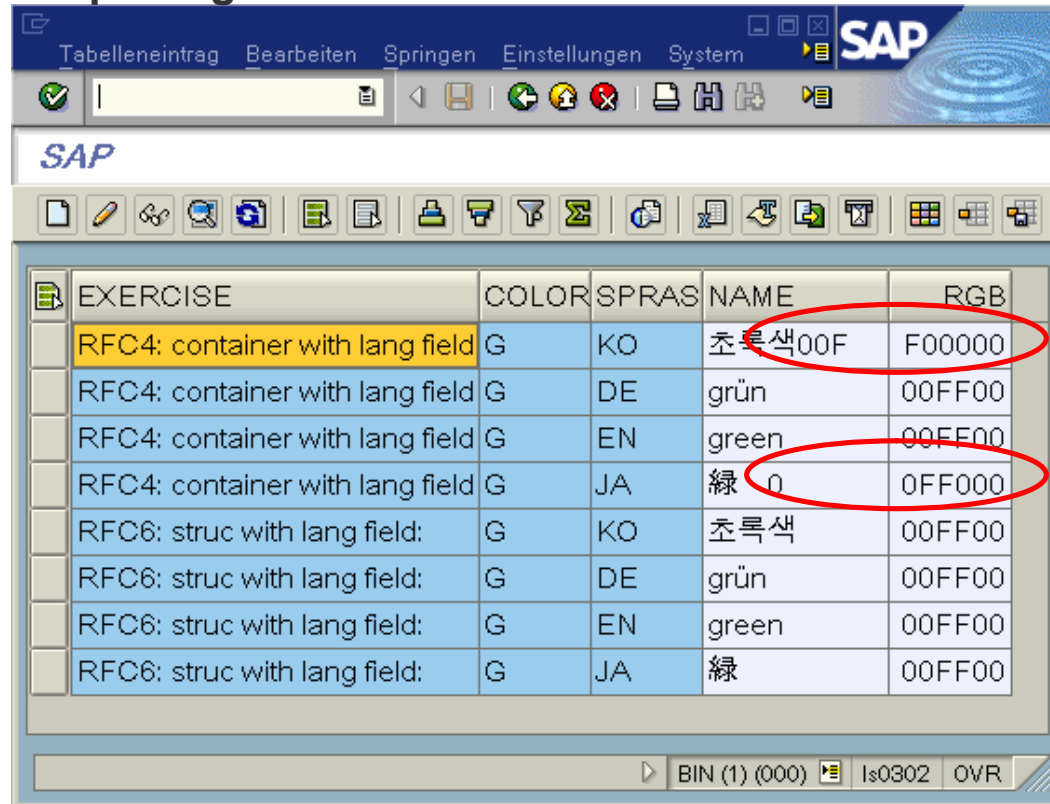
Exercise	COLOR	Lan...	NAME	RGB
RFC2: container with lang field	G	KO	초록색	200020
RFC2: container with lang field	G	DE	grün	0000FF
RFC2: container with lang field	G	EN	green	0000FF
RFC2: container with lang field	G	JA	緑	CFF800
RFC6: struc with lang field:	G	KO	초록색	00FF00
RFC6: struc with lang field:	G	DE	grün	00FF00
RFC6: struc with lang field:	G	EN	green	00FF00
RFC6: struc with lang field:	G	JA	緑	00FF00

Common mistakes: Type hiding: characterlike data

Don't hide types 2

Even sending a pure characterlike structure in a character container conceals important information – the field boundaries – from the system.

Example: ☠ Transporting characterlike data in character containers



The screenshot shows an SAP table named EXERCISE. The table has five columns: EXERCISE, COLOR, SPRAS, NAME, and RGB. The data is as follows:

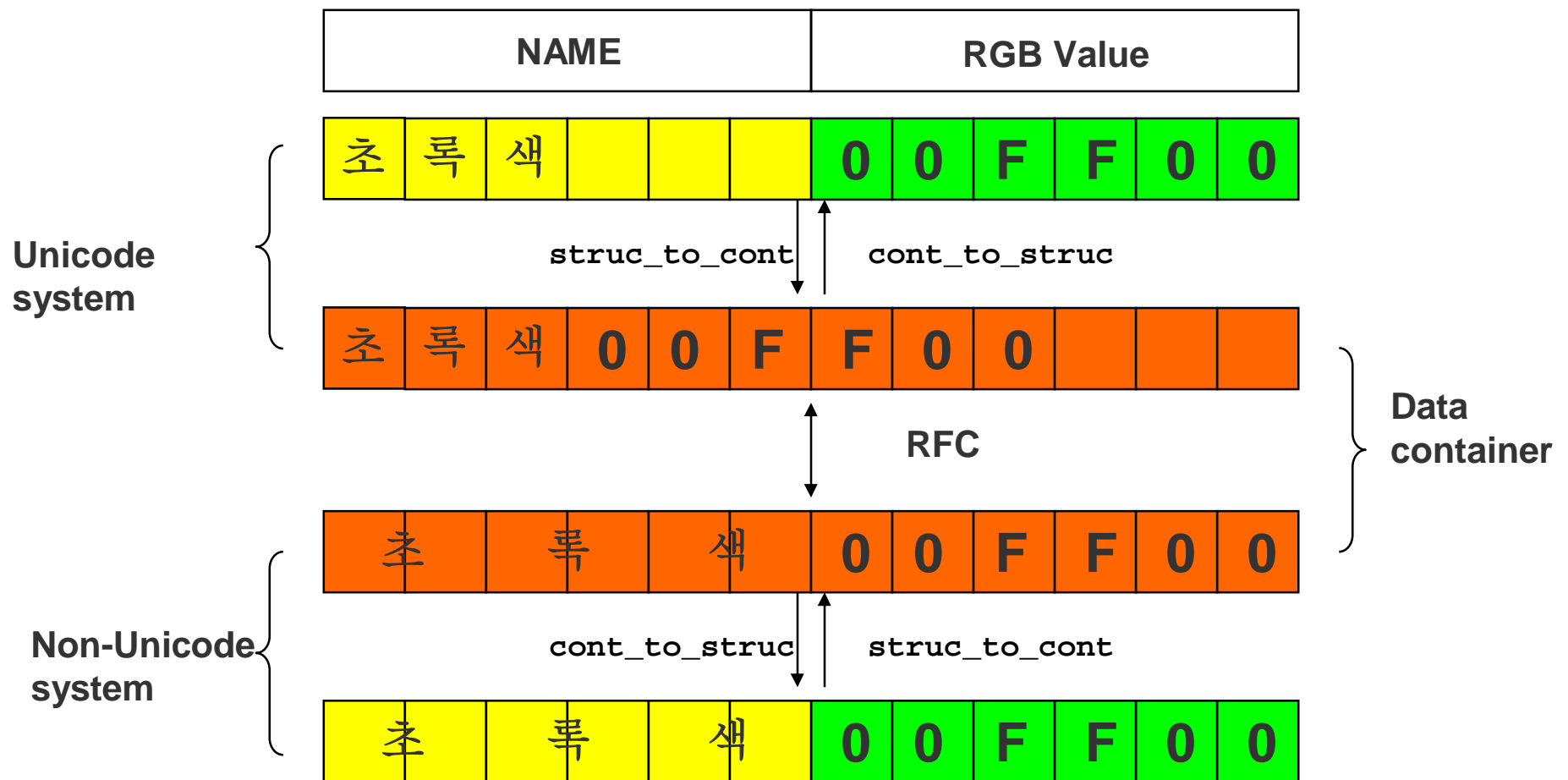
EXERCISE	COLOR	SPRAS	NAME	RGB
RFC4: container with lang field	G	KO	초록색00F	F00000
RFC4: container with lang field	G	DE	grün	00FF00
RFC4: container with lang field	G	EN	green	00FF00
RFC4: container with lang field	G	JA	緑 0	0FF000
RFC6: struc with lang field:	G	KO	초록색	00FF00
RFC6: struc with lang field:	G	DE	grün	00FF00
RFC6: struc with lang field:	G	EN	green	00FF00
RFC6: struc with lang field:	G	JA	緑	00FF00

In the original image, red circles highlight the NAME and RGB columns for the first and fourth rows, showing how the character container obscures the field boundaries. The first row shows '초록색00F' and 'F00000', while the fourth row shows '緑 0' and '0FF000'.

Common mistakes: Type hiding: characterlike data

Workaround if container approach cannot be changed

Use `CL_NLS_STRUC_CONTAINER` to correct the implicit layout:

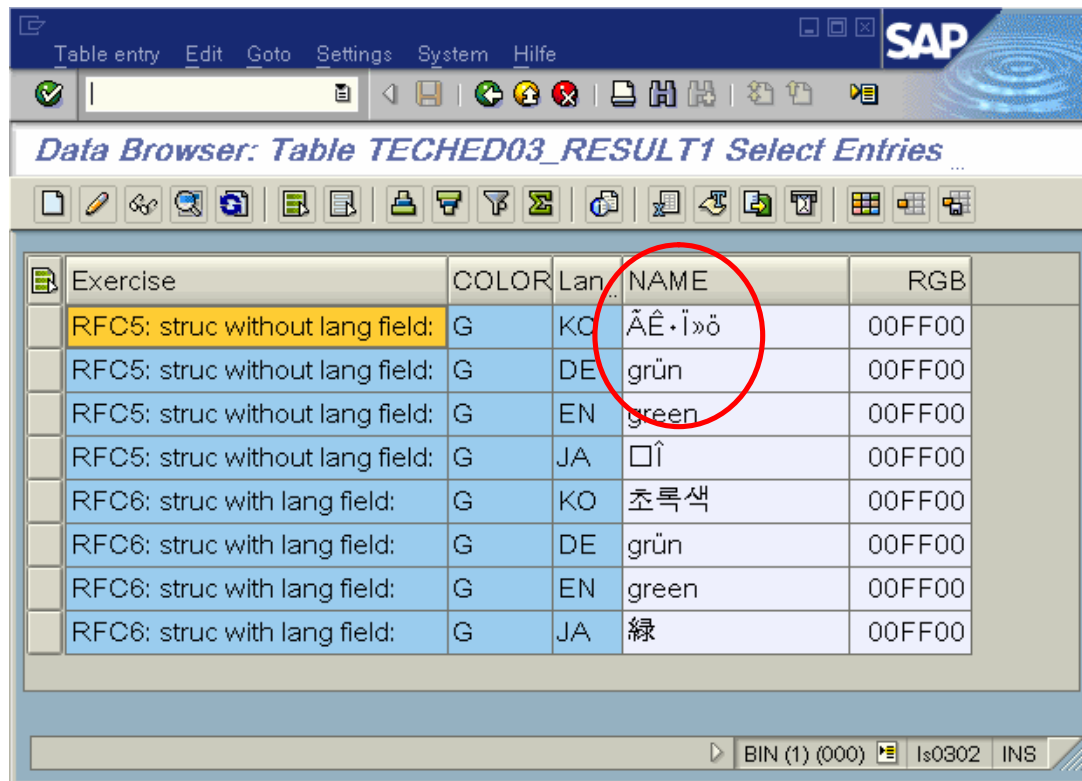


Common mistakes: Missing text language field

Always use text language fields

In principle you must not send any data without text language if the data contains non 7 bit ASCII characters. Otherwise corruption of the data is the result.

Example: ☠ Sending non Latin 1 data without text language field by RFC (German logon)



The screenshot shows the SAP Data Browser interface. The table title is "Data Browser: Table TECHED03_RESULT1 Select Entries". The table has the following columns: Exercise, COLOR, Lan., NAME, and RGB. The first row is highlighted in yellow and circled in red. The text in the NAME field of this row is "ÃÊ·ï»ö", which is a corrupted version of the German word "grün".

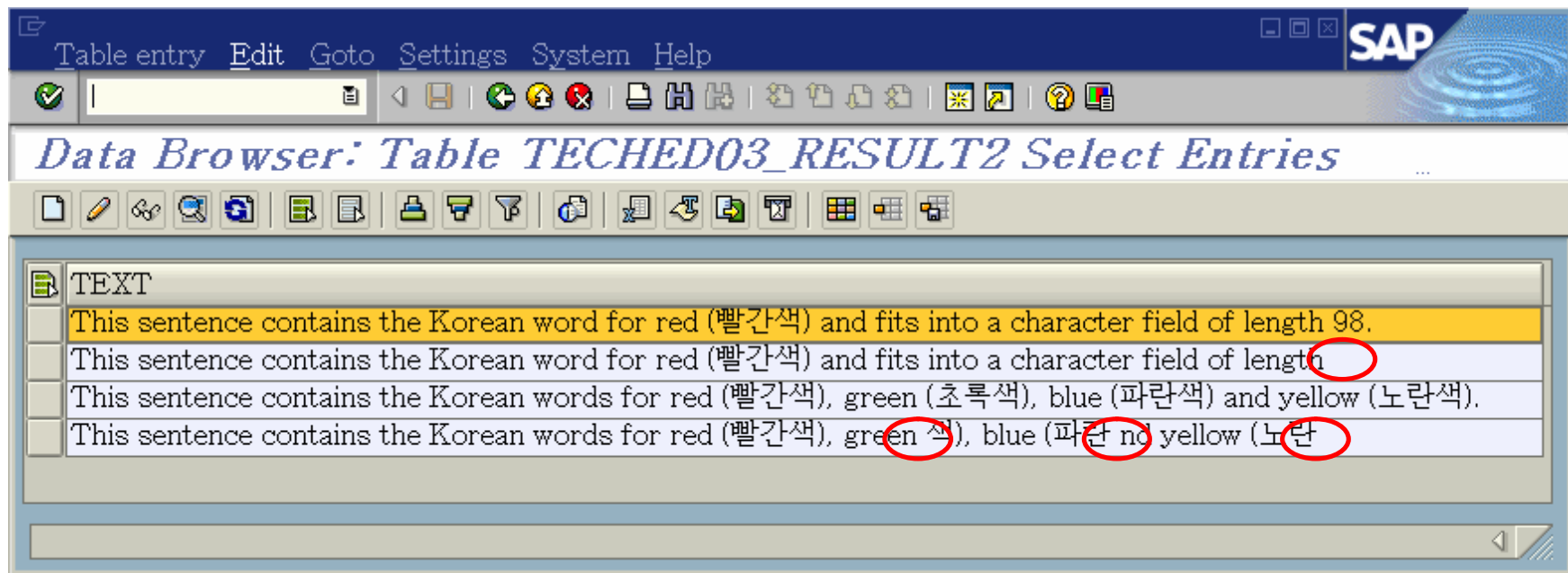
Exercise	COLOR	Lan.	NAME	RGB
RFC5: struc without lang field:	G	KO	ÃÊ·ï»ö	00FF00
RFC5: struc without lang field:	G	DE	grün	00FF00
RFC5: struc without lang field:	G	EN	green	00FF00
RFC5: struc without lang field:	G	JA	□↑	00FF00
RFC6: struc with lang field:	G	KO	초록색	00FF00
RFC6: struc with lang field:	G	DE	grün	00FF00
RFC6: struc with lang field:	G	EN	green	00FF00
RFC6: struc with lang field:	G	JA	緑	00FF00

Common mistakes: Wrong length assumptions

Problems with length assumptions

String lengths are not invariant under code page conversions. This may lead to different problems:

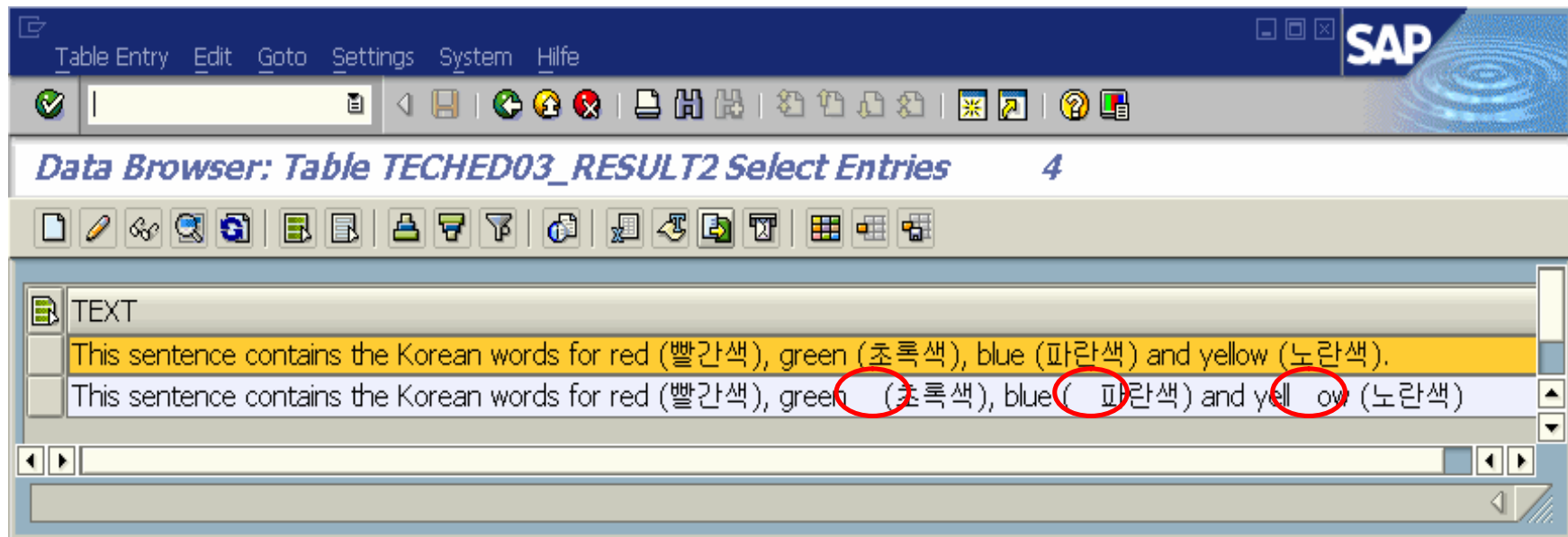
In a Unicode system a character field of certain length can hold more characters than the same character field in a non-Unicode system. Sending such data will result in data loss (☠).



Common mistakes: Wrong length assumptions

Problems with length assumptions (continued)

Breaking a string into a table of fixed line size and sending the table from a non-Unicode to a Unicode-system does not work, since the information about the occupied length is lost and subsequent reassembling into a string will insert unwanted spaces (☹).

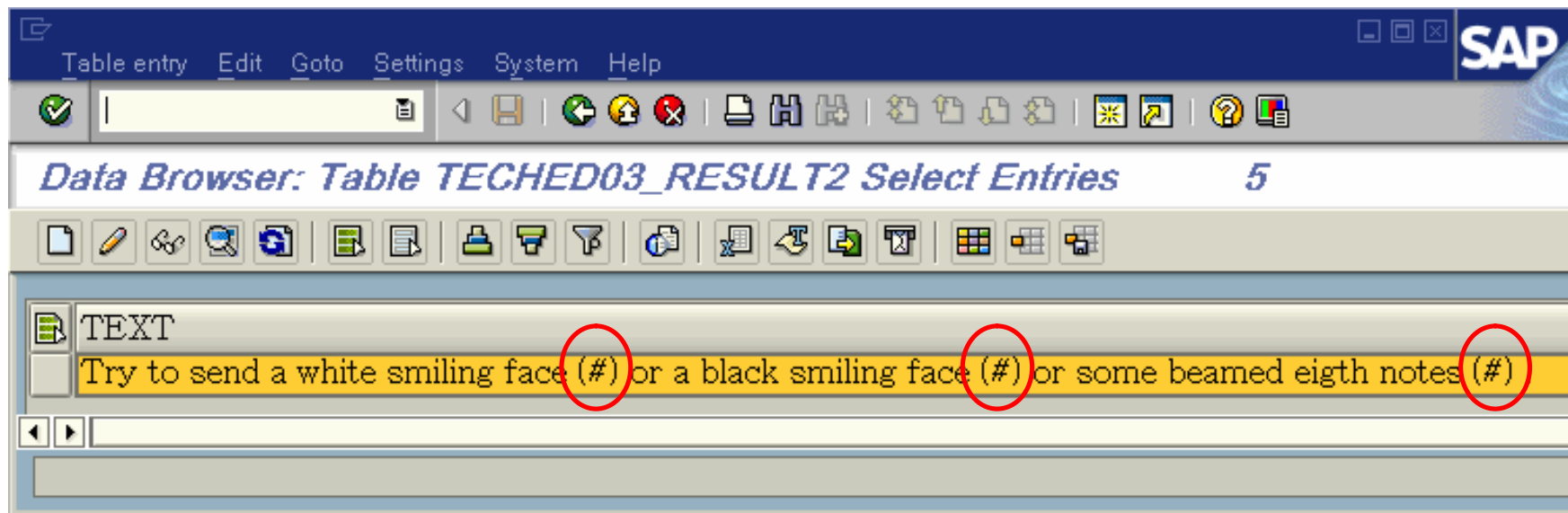


Common mistakes: data not in receivers codepage

Data not in the receivers code page

In general you must not send data from a source system into a target system, if the characters send are not in the target systems code page. Especially don't send one of the characters that are only in the Unicode code page to an old-fashioned non-Unicode system:

Try to send a white smiling face (☺) or a black smiling face (☹) or some beamed eighth notes (♪) ! (→ # → ☹)



Introduction

- About Code Pages
- The Ideal Picture
- Reality

Part I – RFC

- Unicode ↔ Unicode
- Unicode ↔ single code page system
- Unicode ↔ MDMP system

Part II – File transfer

- Writing and reading files on the application server
- Writing and reading files on the front end

Part III – Common mistakes

Exercises

Send single code page and MDMP data via RFC

- Type hiding and missing text language fields:
TECHED_UNICODE_EXERCISE_11/12/13/14 and 15
- Wrong length assumptions:
TECHED_UNICODE_EXERCISE_16/18
- Data not in the receiver's code page:
TECHED_UNICODE_EXERCISE_17

Transfer data via file on the application server

- Writing files: TECHED_UNICODE_EXERCISE_19
- Reading files: TECHED_UNICODE_EXERCISE_20
- BOM handling: TECHED_UNICODE_EXERCISE_23

Transfer data via file on the frontend

- Writing files: TECHED_UNICODE_EXERCISE_21
- Reading files: TECHED_UNICODE_EXERCISE_22

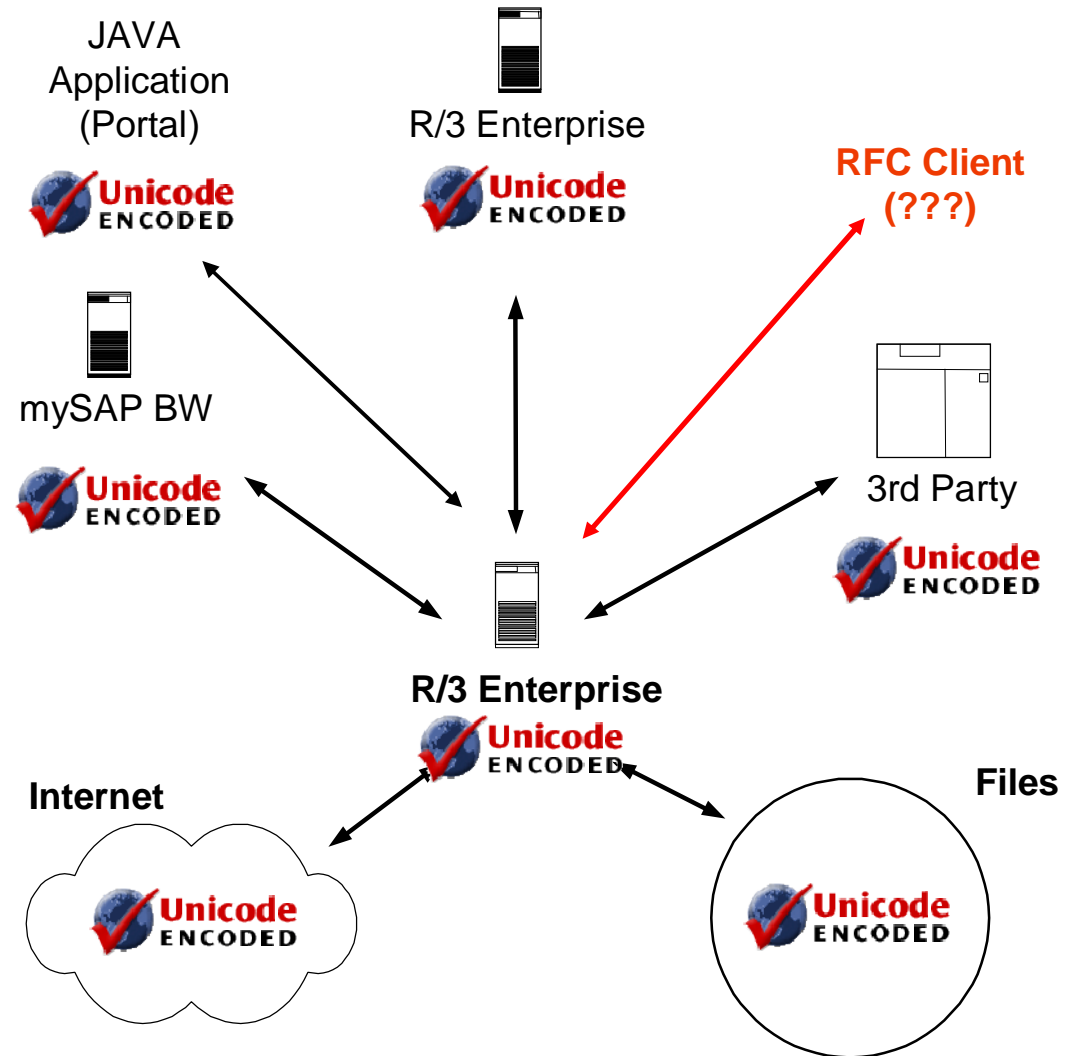
Appendix

- RFC SAP R3 ↔ RFC client: Unicode interface for C programming
- Other languages



The ideal Picture: RFC Clients ???

- When I want to Unicode enable my RFC client ...
- ... what shall I do with type char ???



What you need – Unicode RFC SDK

<http://service.sap.com/netweaver>

SAP NetWeaver

SAP NetWeaver in Detail

Application Platform

Connectivity

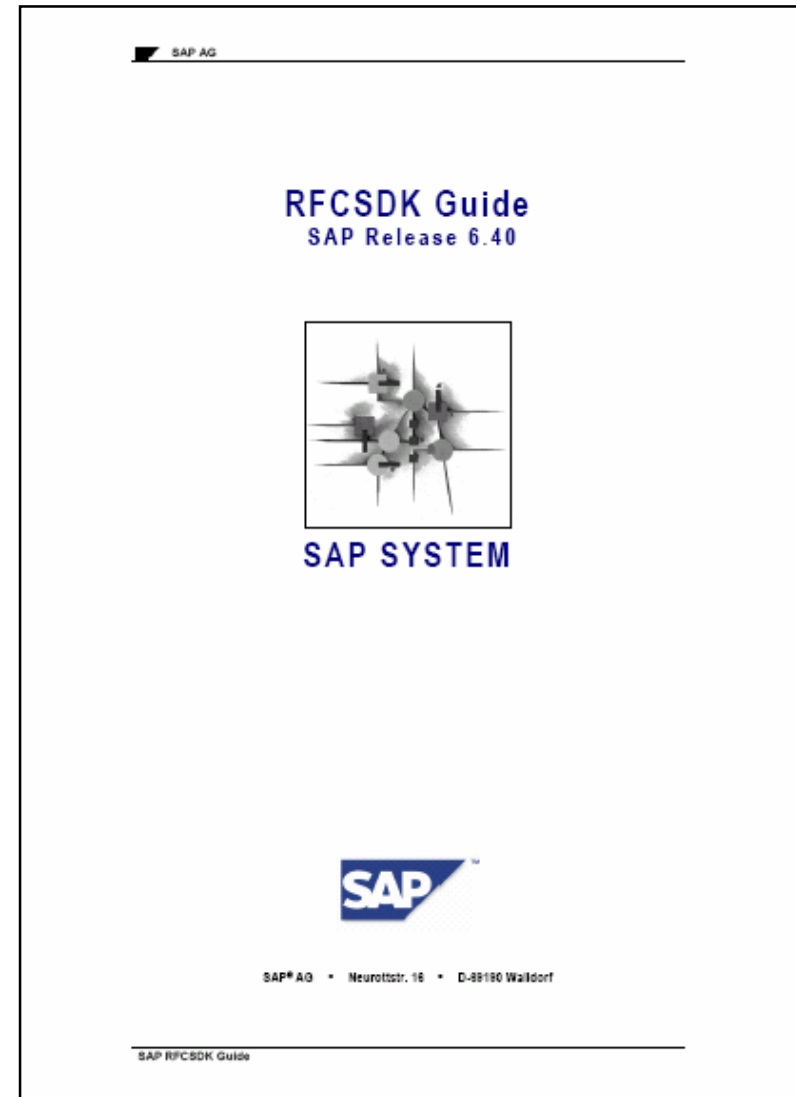
Connectors

RFC Library

Media Library

RFC Library Guide (PDF)

Section: RFC Library and UNICODE



Java uses Unicode → Unicode R/3 makes live easier

For JCo server it is necessary to specify if partners are Unicode:

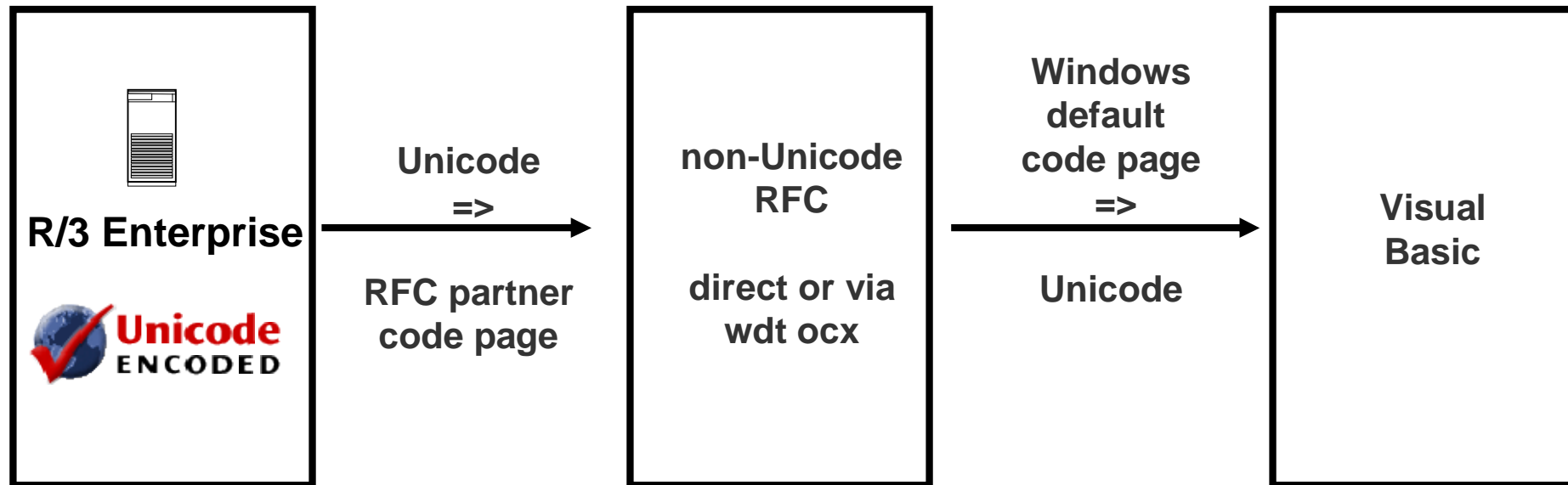
```
JCo.Server.setProperty("jco.server.unicode", "1");
```

JCo server needs different repository for Unicode and non-Unicode partners

→ One JCo.Server Object can only serve Unicode or non-Unicode partners

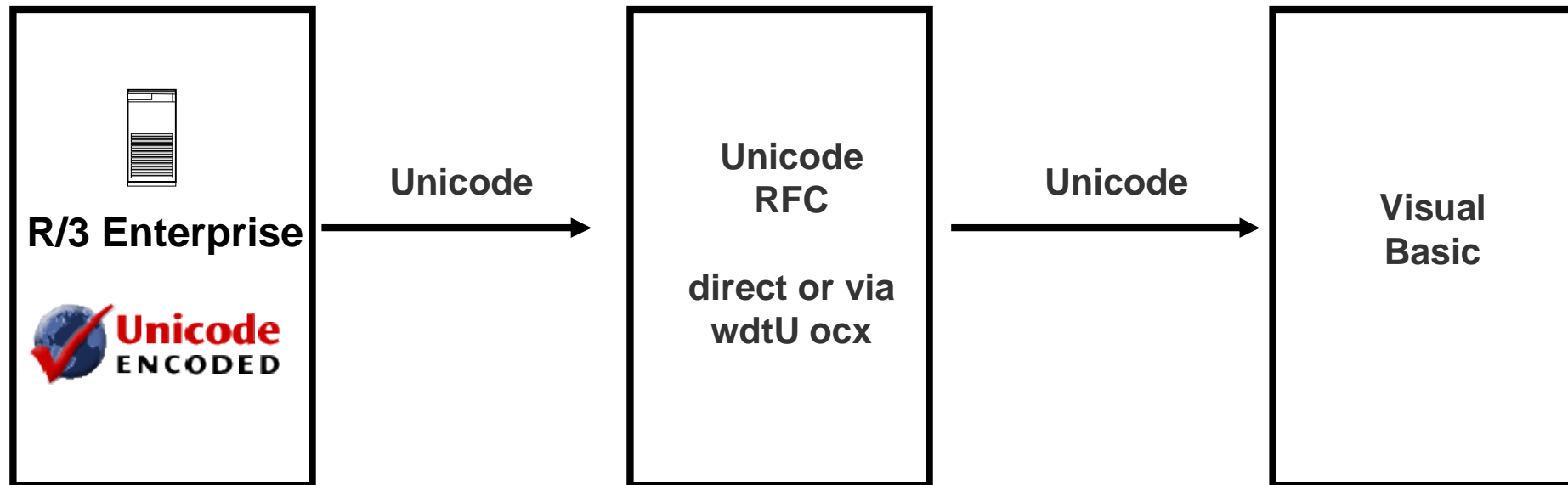
This restriction does not exist for JRFC (JCo successor for J2EE applications)

Communication with Visual Basic (1)



Data transfer is restricted to Windows default code page

RFC partner code page must match to Windows default code page



With Unicode RFC / Unicode wdt ocx no code page conversion needed

**Standard Visual Basic UI controls do not support Unicode
MS Office controls do**

Visual Basic .NET uses Unicode

Via .NET Connector communication with Unicode and non-Unicode partners possible (transparent for caller)

Further Information

→ Public Web:

<http://www.service.sap.com/Unicode@sap> : Technology

<http://www.service.sap.com/Unicode> : Customer contact

www.sdn.sap.com/sdn/developersguide.sdn : NetWeaver Developer's Guide

→ Related SAP Education Training Opportunities

<http://www.service.sap.com/rkt-unicode>

→ Related Workshops/Lectures at SAP TechEd 2005

SPC202 Conversion of SAP Systems to Unicode, lecture

SPC204 Performance Techniques for Unicode Conversion of Single Code Page Systems, lecture (TechEd Boston only)

LCM207 SAP Upgrade and Unicode Methodology, (TechEd Boston only)

SPC251 Unicode Interfaces – Data Exchange Between Unicode and non-Unicode Systems, hands on

SPC250 Making ABAP Programs Unicode Enabled; hands on

IM101 Dealing with Multi-Language Garbage Data – Lessons Learned, lecture

Questions?



Q&A



Feedback

Please complete your session evaluation.

Be courteous — deposit your trash,
and do not take the handouts for the following session.

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, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
 - IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation.
 - Oracle is a registered trademark of Oracle Corporation.
 - 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.
 - Java is a registered trademark of Sun Microsystems, Inc.
 - JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.
 - MaxDB is a trademark of MySQL AB, Sweden.
 - SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, 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 in several other countries all over the world. 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.
 - This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.
 - SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. 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 shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.
 - The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.