UNICODE und IBM i

Multilinguale Einsatzbereiche sind heute keine Seltenheit mehr. Die globale Verfügbarkeit eines Systems öffnet neue Märkte, stellt aber immer wieder massive Anforderungen an die IT. Kompatibilität, Unterstützung von Zeichensätzen und die Speicherung der Daten in einer für alle Anwendungen zu verarbeitenden Form sind nur einige der Punkte, die bei einem globalen Einsatz in Bezug auf die Speicherung und Darstellung der Informationen bedacht werden müssen.

Wie einfach waren die Anfangszeiten der IT. Zentrale Verarbeitung und klar definierte Zeichen, die anzuzeigen, zu drucken oder zu speichern waren. Doch je globaler sich im Laufe der Jahre die Märkte entwickelt haben, desto schneller wuchsen auch die Anforderungen an die IT, die vielfältigen Formen der Speicherung von Daten oder besser deren Bestandteilen in Form der Zeichen zu bewerkstelligen.

Haben Sie sich einmal Gedanken gemacht, wie beispielsweise unser ,“Ä“ oder „ß“ in Japan dargestellt werden? Haben Sie einmal französische oder tschechische Daten zur Verarbeitung erhalten? Hier beginnen die Auswirkungen der Globalisierung die Grenzen der Standard-IT auszuhebeln.

Wenn Daten nur in einer Sprache (gemeint ist hier wirklich die Sprache und nicht etwa eine Programmiersprache!) bereitgestellt werden müssen, dann kann man dies mit den klassischen Zeichensätzen durchführen, die eine besondere Ausrichtung auf die Unterstützung der nationalen Zeichen haben. Sollen aber mehrere unterschiedliche Sprachen mit abweichenden Zeichensätzen verarbeitet werden, dann müssen unterschiedliche Zeichensätze oder Codepages eingesetzt werden.

Zeichensätze sind die Grundlage für die Darstellung der Daten, die mit einem beliebigen IT System gespeichert werden. So unterschiedlich die einzelnen Zeichensätze auch sein mögen - allen basieren auf der Speicherung der Informationen in Form von numerischen Werten. So wird zum Beispiel der Buchstabe „A“ als Wert 0100 0001 dargestellt.

Leider haben sich im Laufe der Jahre unterschiedliche Standards entwickelt, die zum Teil erhebliche Unterschiede aufweisen. Kompatibilität und auch Möglichkeiten der Zeichendarstellungen der Zeichensätze bestimmen neben den Anwendungen, welcher Zeichensatz verwendet werden kann oder verwendet werden muss. Limitierungen, die bei der Entwicklung der älteren Zeichensätze noch nicht bedacht wurden, sind heute häufig Grund für Anpassungen. Die Unterstützung mehrerer Sprachen beispielsweise – heute eine Normalanforderung in den meisten Unternehmen – ist nicht oder nur teilweise möglich.

Nicht unproblematisch sind dann die Fehler bei den möglichen Transformationen, die bei der Umsetzung von einem Zeichensatz in einen anderen Zeichensatz vorgenommen werden müssen. „Wundersame“ Zeichen sind dann die Folge, die bis zur Verstümmelung des Ursprungstextes führen kann.

Sicher sind Ihnen auch einige der folgenden Zeichensätze bekannt:

ASCII
Mit diesem auf Einzelbytezeichen basierten Codierschema können je nach verwendetem Bit-Einsatz 128 bzw. bis zu 256 Zeichen dargestellt werden. Mit 7 Bits werden alle im englischen Alphabet vorkommenden Buchstaben, Ziffern, Satz- und Steuerzeichen dargestellt. Mit einer 8 Bit Umsetzung können zusätzliche Sonderzeichen dargestellt werden.

EBCDIC
Dieser von IBM für Großrechner entwickelte Zeichensatz dominiert die IBM Systeme. EBCDIC ist die Abkürzung für Extended Binary Coded Decimals Interchange Code. Dieser Zeichensatz basiert auf einem 8-Bit Code.
Eine weitere Unterscheidung findet mit der Codepage statt. IBM bietet 14 verschiedene Codepages, mit denen jeweils die landes – oder besser sprachenspezifischen Anforderungen an die Darstellungen von Zeichen abgedeckt werden.

Hier eine Übersicht der Codepages:

- 00037 USA, Kanada
- 00273 Deutschland, Österreich
- 00277 Dänemark, Norwegen
- 00278 Finnland, Schweden
- 00280 Italien
- 00284 Spanien, Südamerika
- 00285 Großbritannien
- 00297 Frankreich
- 00423 Griechenland
- 00500 Belgien, Schweiz, Kanada
- 00838 Thai

Sicher kennen Sie diese durch die Einstellungen auf Ihrem System i. Hier kann man beispielsweise über den Systemwert QCHRID die Codepage festlegen. Diese hat zum Teil erhebliche Auswirkungen, wie die beiden nachfolgenden Codepages zeigen:

Die EBCDIC Codepage 273, die meist für den deutschsprachigen Raum eingesetzt wird:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	
4_			â	{	à	á	ã	å	ç	ñ	Ä	.	<	(	+	!	**4_ (4 hex = 0100 bin)**
5_	&	é	ê	ë	è	í	î	ï	ì	~	Ü	$	*	)	;	^	**5_ (5 hex = 0101 bin)**
6_	-	/	Â	[	À	Á	Ã	Å	Ç	Ñ	ö		%	_	>	?	**6_ (6 hex = 0110 bin)**
7_	ø	É	Ê	Ë	È	Í	Î	Ï	Ì	`	:	#	§	'	=	"	**7_ (7 hex = 0111 bin)**
8_	Ø	a	b	c	d	e	f	g	h	i	«	»	ð	ý	þ	±	**8_ (8 hex = 1000 bin)**
9_	°	j	k	l	m	n	o	p	q	r	ª	º	æ	¸	Æ	¤	**9_ (9 hex = 1001 bin)**
A_	µ	ß	s	t	u	v	w	x	y	z	¡	¿	Ð	Ý	Þ	®	**A_ (A hex = 1010 bin)**
B_	¢	£	¥	·	©	@	¶	¼	½	¾	¬		¯	¨	´	×	**B_ (B hex = 1011 bin)**

C_	ä	A	B	C	D	E	F	G	H	I		ô	¦	ò	ó		**C_ (C hex = 1100 bin)**
D_	ü	J	K	L	M	N	O	P	Q	R	¹	û	}	ù	ú	ÿ	**D_ (D hex = 1101 bin)**
E_	Ö	÷	S	T	U	V	W	X	Y	Z	²	Ô	\	Ò	Ó	Õ	**E_ (E hex = 1110 bin)**
F_	0	1	2	3	4	5	6	7	8	9	³	Û	]	Ù	Ú		**F_ (F hex = 1111 bin)**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	

Für die Codepage 37 (USA, Australien etc) werden die folgenden hexadezimalen Werte genutzt:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	
4_			â	ä	à	á	ã	å	ç	ñ	¢	.	<	(	+		**4_ (4 hex = 0100 bin)**
5_	&	é	ê	ë	è	í	î	ï	ì	ß	!	$	*	)	;	¬	**5_ (5 hex = 0101 bin)**
6_	-	/	Â	Ä	À	Á	Ã	Å	Ç	Ñ	¦		%	_	>	?	**6_ (6 hex = 0110 bin)**
7_	ø	É	Ê	Ë	È	Í	Î	Ï	Ì	`	:	#	@	'	=	"	**7_ (7 hex = 0111 bin)**
8_	Ø	a	b	c	d	e	f	g	h	i	«	»	ð	ý	þ	±	**8_ (8 hex = 1000 bin)**
9_	°	j	k	l	m	n	o	p	q	r	ª	º	æ	¸	Æ	¤	**9_ (9 hex = 1001 bin)**
A_	µ	~	s	t	u	v	w	x	y	z	¡	¿	Ð	Ý	Þ	®	**A_ (A hex = 1010 bin)**
B_	^	£	¥	·	©	§	¶	¼	½	¾	[	]	¯	¨	´	×	**B_ (B hex = 1011 bin)**
C_	{	A	B	C	D	E	F	G	H	I		ô	ö	ò	ó		**C_ (C hex = 1100 bin)**
D_	}	J	K	L	M	N	O	P	Q	R	¹	û	ü	ù	ú	ÿ	**D_ (D hex = 1101 bin)**
E_	\	÷	S	T	U	V	W	X	Y	Z	²	Ô	Ö	Ò	Ó	Õ	**E_ (E hex = 1110 bin)**
F_	0	1	2	3	4	5	6	7	8	9	³	Û	Ü	Ù	Ú		**F_ (F hex = 1111 bin)**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	

UTF-8

Hier handelt es sich um eine Erweiterung des ASCII Zeichensatzes mit 16-Bit Unicode Nutzung. Damit können weitaus mehr Zeichen dargestellt werden, als es der „alte“ ASCII Zeichensatz gestattet.

Dies ist nur eine kleine Übersicht zu dem Thema Zeichensätze. Darüber hinaus existieren noch einige weitere Zeichensätze, die wir hier nicht weiter behandeln wollen.

Beispielszenario

Nehmen wir einmal ein Alltagsbeispiel für die Zeichensätze und die damit verbundenen Probleme:

Währungskennzeichen sind heute im internationalen Handel wichtig und fast in jedem Unternehmen auf Ausdrucken oder Anzeigen zu finden.

Wie wird nun ein solches Währungskennzeichen in den zuvor genannten Zeichensätzen definiert? Schauen wir uns das am Beispiel des Dollarzeichens ($) an:

Bei der Verwendung der EBCDIC Codepage 37:wird das Dollarzeichen ($) mit dem hexadezimalen Wert 5B dargestellt.

Wird hingegen die Codepage 285 verwendet, dann finden wir als hexadezimalen Wert für das Dollarzeichen($) 4A. 5B wird hier für das Währungszeichen des GBP verwendet.

Für den in Deutschland gebräuchlichen Zeichensatz 273 ist das Dollarzeichen wieder mit dem hexadezimalen Wert 5B dargestellt.

Was passiert nun, wenn Daten mit der Codepage 285 verarbeitet werden? Die Antwort ist schlicht – es wird der hier gefundenen hexadezimale Werte dargestellt.

Wie Sie sehen, ist der Einsatz der Codepage mit Vorsicht zu genießen und man muss sehr wohl überlegen, ob man diese Form der Zeichensatzunterstützung weiter betreibt, wenn die Komplexität der Unterstützung verschiedener Sprachen zu groß wird.

Die Verwendung der Zeichensätze über Plattformen oder Anwendungen hinweg ist nicht trivial. Im Idealfall verwenden unterschiedliche Systeme denselben Zeichensatz. Anderenfalls sind unter Umständen Umsetzungen durchzuführen. Diese sind zwar auch kein Hexenwerk, bedürfen aber zusätzlicher Rechenzeit und stellen so gerade bei Massendatenverarbeitungen ein gewisses Hindernis dar.

Unicode

Damit die unterschiedlichen Informationen und Zeichen einheitlich verarbeitet werden können, hat man einen allgemeingültigen Standard entwickelt, mit dem die alphanumerischen Zeichen einheitlich dargestellt werden können. „Unicode“ (Einheitsschlüssel) dient damit der einheitlichen Darstellung und Speicherung von Buchstaben, Ziffern und Sonderzeichen. Damit hat man erstmals relativ erfolgreich den Versuch unternommen, alle bekannten Textzeichen aus den verschiedensten Quellen in einem Zeichensatz darstellen zu können.

Unicode bietet sich besonders dann an, wenn Ihre Daten in unterschiedlichsten Sprachen (mit verschiedenen Zeichensätzen) abgebildet werden müssen.

Das zuvor gezeigte Beispiel würde mit Unicode nicht zu einem Problem führen, da die Zeichendarstellung eindeutig ist. Mit Unicode wird das Dollarzeichen ($) beispielsweise mit dem hexadezimalen Wert 0024 und das Pfundzeichen mit dem Hexwert 00A3 dargestellt. Durch diese Eindeutigkeit muss man sich in den Anwendungen nicht um Zeichensatzprobleme und Umsetzungen kümmern.

Damit die Vielfältigkeit der darzustellenden Zeichen mit einem einzelnen Zeichensatz abgedeckt werden kann, wird eine maximale 4 Byte Referenz (UTF32) für die Zeichendarstellung verwendet.

Verantwortlich für die Erstellung und die Pflege des Unicodes ist ein Konsortium, in dem unter anderen auch die führenden IT-Unternehmen beteiligt sind. Das Ziel dieses gemeinnützigen Konsortiums liegt darin, die Darstellung von Textdaten mit unterschiedlichen IT Systemen zu vereinheitlichen.

Die eigentliche Speicherung erfolgt in binärer Form, die definiert, welches Zeichen in welchem Byte-Wert definiert ist. Aber was wäre ein Standard, wenn es nicht auch noch andere Standards gäbe? Für den Einsatz mit dem Internet hat man ebenfalls einen global einsetzbaren Standard in Form der Codierung im ASCII Format entwickelt. In ASCII erfolgt die Speicherung der Zeichen natürlich auch im Byteformat. Dabei wird jedes Zeichen mit 7 Bits definiert. Folglich sind maximal 128 Zeichen unterstützt.

Aber wir wollen hier nicht zu sehr in die Details der Zeichensätze abweichen und uns dem Unicode wieder zuwenden:

Wann sollte man sich denn nun eigentlich Gedanken über den Einsatz von Unicode machen?
Diese Frage lässt sich am ehesten mit den folgenden Argumenten beantworten:

1. Wenn Anwendungen plattformunabhängig ausgeführt werden sollen.
2. Wenn die Daten von unterschiedlichsten Anwendungen verarbeitetet werden müssen (z.B. mit IBM i Anwendungen und Microsoft-Anwendungen).
3. Wenn unterschiedlichste Sprachen genutzt werden.

IBM hat in den letzten Jahren mehr und mehr die Unterstützung von Unicode für i5/OS und IBM i vorangetrieben. Andere Globalplayer der IT setzen bereits seit längerer Zeit auf den Einsatz von Unicode. Sun, Apple, Oracle und Microsoft haben die Vorteile dieser Zeichendarstellung erkannt. Auch XML, HTML und Java Anwendungen „sprechen“ Unicode.

Die Standardfunktionen des IBM i wie zum Beispiel der 5250 Datenstrom, Druckfunktionen und auch die mit WebSphere bereitgestellten Produkte sind heute in der Lage, Unicode zu unterstützen. Dabei spielt die CCSID (Coded Character Set Identifier) des jeweiligen Systems eine entscheidende Rolle.

Mit dem Einsatz von Unicode auf dem IBM i sind wir heute in der Lage, unabhängig von der verwendeten Plattform, den Anwendungen oder auch den Sprachen, in denen die Anwendungen erstellt worden sind, die Informationen verarbeiten zu können.

Doch wozu Unicode? Allgemein gesagt, bietet Unicode die Verwendung unterschiedlichster Zeichen, die von den verschiedensten Sprachen benötigt werden. Diese Verwendung ist dann mit nur einem einzigen Zeichensatz möglich. Das diese aber eine nicht ganz unkomplizierte Sache ist, zeigen die verschiedenen Berührungspunkte, die wir bei dem Einsatz oder gar der Änderung von Zeichensätzen bedenken müssen.

Bei dem Einsatz von Unicode müssen wir nicht nur den nativen IBM i Bereich betrachten, sondern zudem auch die Notwendigkeit im Hinterkopf behalten, dass gegebenenfalls die Daten, welche in der DB2/UDB des IBM gespeichert werden, auch mit nicht IBM Anwendungen verarbeitet werden können müssen. Gerne wird in diesem Zusammenhang die Kompatibilität mit Microsoftprodukten von zum Beispiel Excel erwähnt. Bei dem Datenaustausch muss generell darauf geachtet werden, dass die Informationen in einem unterstützen Format übertragen und am Ziel auch korrekt dargestellt werden. UTF-8 oder auch UTF-16 gehören unweigerlich mit zur Aufzählung der Punkte, die bei dem Einsatz von Unicode zu erwähnen sind.