Using Win32® To Create International Applications

Ken Fowles
Manager Globalization Evangelism
Developer Relations Group
Microsoft Corporation

Agenda

- Chicago international product
- Win32 API for international
- Win32 input method editors
- What to do

Windows *"Chicago"* International

- Same language versions as for Windows™ 3.1
- US/European targets 386/4 MB
- Far East: U.S. requirements + a couple megs
- Win32, fully preemptive multitasking, does not require MS-DOS® separately
- Better international support

Where Windows *"Chicago"* Will Be

When Is International Windows *"Chicago"*?

- U.S. ships second half 1994
- Major European languages sim-ship or close to sim-ship
- Many Far East and Mid East languages within 3-6 months after U.S.

Ship With Win32 Right Now

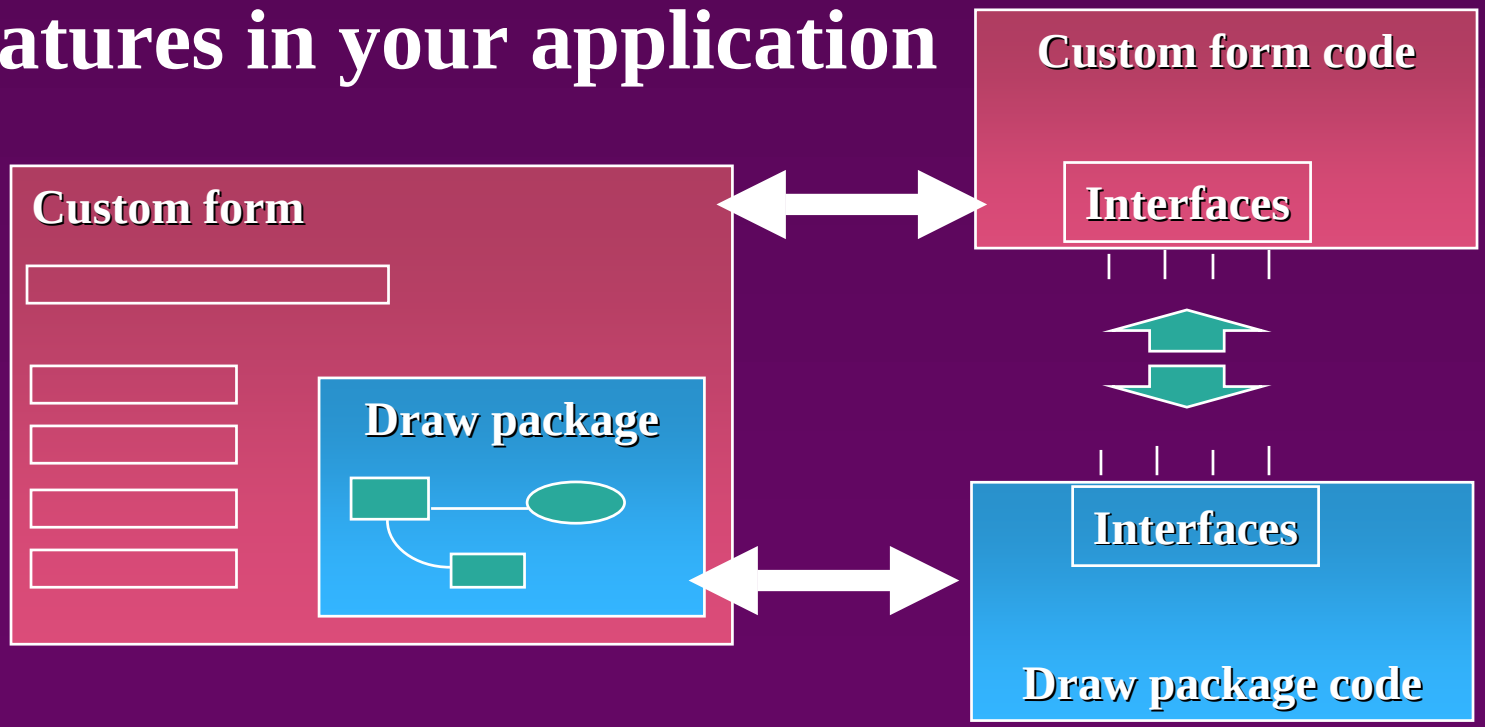
- Prepares you for “Chicago”
- Ship today using Win32s™
  - Run on Windows 3.1, free run-time libraries
  - Run on Windows NT™ as a native 32-bit application
  - Japan version of Win32s shipped
- Performance and capacity improvements
  - 1/2 GB linear address space, no segmentation headaches
- Limitations of Win32s: shared memory, no threads, base Win32 API set

Improved International Services In Windows *"Chicago"*

- **Win32 API**
  - **NLS APIs**
  - **Improvements to Far East IMEs**
  - **Pen recognition including Japanese**
  - **Better graphics and multimedia**
  - **WOSA, messaging, telephony services**
  - **DBCS-enabled ODBC for Japan**
- **Full set of object-oriented services, that is, OLE technology**

OLE 2.0
Works well for international

- Application glue
- Shortcut to country-specific features in your application



OLE International

- Win16 with OLE 2.01 libraries DBCS-enabled, including in the U.S.
- Win32 with OLE 2.01 libraries consistently Unicode™-enabled, including in the U.S.

Script Support In Windows *"Chicago"*

- ANSI APIs (like Windows 3.1)
- Unicode data supported through Win32 conversion APIs
- Display of multiple scripts
  - **SBCS:** Americas, Western Europe, Eastern Europe, Greece, Turkey, Russia
  - **DBCS:** One Far East language at a time, plus all SBCS languages (above)
  - **Bidirectional:** All Middle East languages, plus all SBCS languages (above)

European Script Support

- Large fonts
  - 600+ glyph European subset
  - Covers the six character sets (code pages) used in Europe and the Americas
  - Covers the associated MS-DOS code pages used in Europe and the Americas
  - Arial, Times New Roman, Courier
- Keyboard layouts
- Locale data for the NLS APIs

Multilingual And NLS API Support

Release	ANSI NLS APIs	Wide NLS APIs	Multi-lingual
Windows NT 3.1	No	Yes	No
Windows NT 3.5	Yes	Yes	No
“Chicago”	Yes	No	Yes
“Cairo”	Yes	Yes	Yes

Agenda

- “Chicago” international product
- **Win32 API for international**
- Win32 input method editors
- What to do

Win32 NLS APIs
"Chicago" and Windows NT

- Locale-based
  - Unique set of regional settings
- APIs for retrieving international data
  - Individual locale elements
  - Composite date and time formats
  - Composite currency and number formats
- APIs for string comparison
- APIs for string conversion
- APIs for other string processing

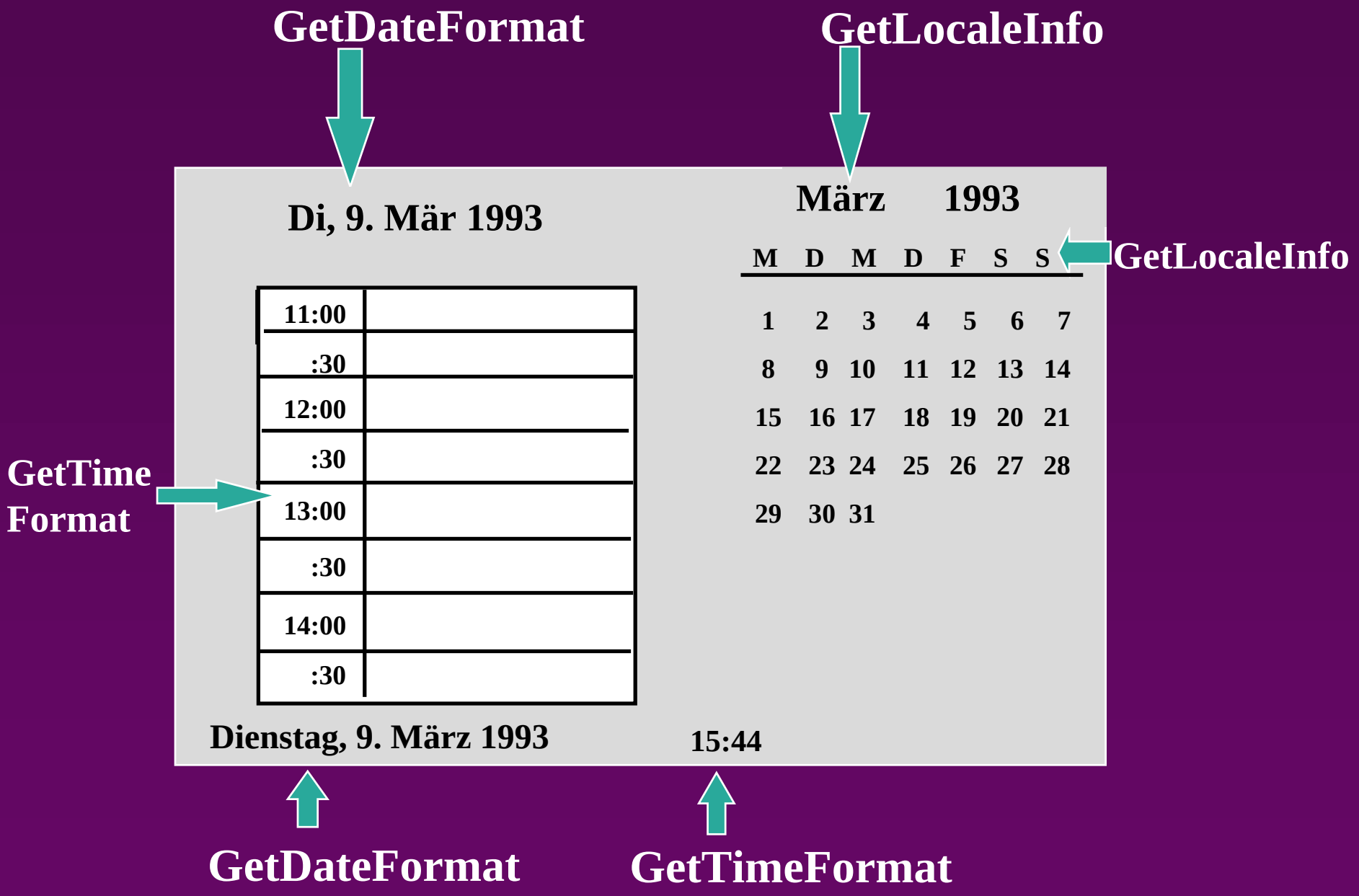
Win32 Locale ID (LCID)

- Comprised of:
  - Primary language
  - Sublanguage
  - Sort index
- Uniquely defines a locale:

MakeLangID (LANG_FRENCH,
SUBLANG_FRENCH_CANADIAN)

NLS APIs

- International data and formats
  - GetLocaleInfo[W|A]
  - SetLocaleInfo[W|A]
  - GetTimeFormat[W|A]
  - GetDateFormat[W|A]
  - EnumDateFormats[W|A]
  - EnumTimeFormats[W|A]
  - EnumCalendarInfo[W|A]
  - GetCurrencyFormat[W|A]
  - GetNumberFormat[W|A]


GetDateFormat
GetLocaleInfo
Di, 9. Mär 1993
März 1993
M D M D F S S
GetLocaleInfo
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
GetTime Format
11:00
:30
12:00
:30
13:00
:30
14:00
:30
Dienstag, 9. März 1993
15:44
GetDateFormat
GetTimeFormat

NLS APIs

- Sorting, string comparison, and mapping
  - LCMapString[W|A]
  - CompareString[W|A]
  - FoldStringW
- Character/string typing
  - GetStringType[W|A]
- Miscellaneous locale information
  - IsValidLocale
  - ConvertDefaultLocale
  - EnumSystemLocales[W|A]

English:

andere
ändere
Chomsky
cote
coté
côte
côté
Käse
Kashubian
map
måp
Masse
Maße
my
piña
pint
søg
strå
straight
thorn
þorn
zoo

French:

andere
ändere
Chomsky
cote
côte
coté
côté
Käse
Kashubian
map
måp
Masse
Maße
my
piña
pint
søg
strå
straight
thorn
þorn
zoo

Norwegian:

andere
Chomsky
cote
coté
côte
côté
Kashubian
Käse
map
Masse
Maße
my
måp
piña
pint
straight
strå
søg
thorn
zoo
þorn
ändere

Spanish:

andere
ändere
cote
coté
côte
côté
Chomsky
Käse
Kashubian
map
måp
Masse
Maße
my
pint
piña
søg
strå
straight
thorn
þorn
zoo

String Matching

Address Book

Angel Peña
Astrid Ångström
Christiane Schönstein
Elisabeth Schonstein
Gérard Laurent
Jörg Straßer
Jørn Heger
José Mendez
Ólafur Asmundsson
Sylvia Strasser

Search for: jor

Found:
Jörg Straßer
Jørn Heger

Search for: strass

Found:
Jörg Straßer
Sylvia Strasser

CompareString using LCID=Standard French
NORM_IGNORECASE
NORM_IGNORENONSPACE

NLS APIs And Scripts

- Code page information
  - IsValidCodePage
  - EnumSystemCodePagesW
  - GetACP
  - GetOEMCP
  - GetCPInfo
  - IsDBCSLeadByte
- Code page conversion
  - MultiByteToWideChar
  - WideCharToMultiByte

Use The Win32 NLS APIs

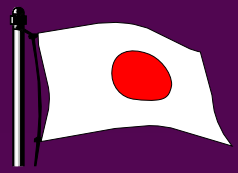
- Accurate international data and sorting
- Easier to use
  - Shortcut to greater functionality
- Lower development costs
  - Eliminates need for proprietary sorting
  - Avoid messing with WIN.INI
- Documented in Windows NT
- Available in “Chicago” and Windows NT

Agenda

- “Chicago” international product
- Win32 API for international
- **Win32 input method editors**
- What to do

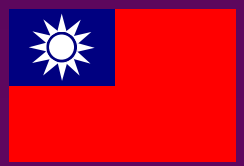
Input Method Editors

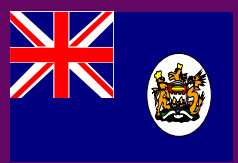
- Windows 3.x and Windows NT 3.x
  - Different IME API set per language
  - Supports single IME context per system
- “Chicago” has a new IME architecture
  - Common API across the Far East markets
  - Provides IME class and IME UI window
  - Supports multiple IME context



IME Support For Windows *"Chicago"*

- Three levels of IME support for applications
  - 1: Global IME class/context
    - For applications that are unaware of IMEs
  - 2: Application IME class/context
    - Easiest for IME-aware applications to implement
  - 3: Direct IME APIs for applications
    - Most control for applications
- Multiple IME context per application

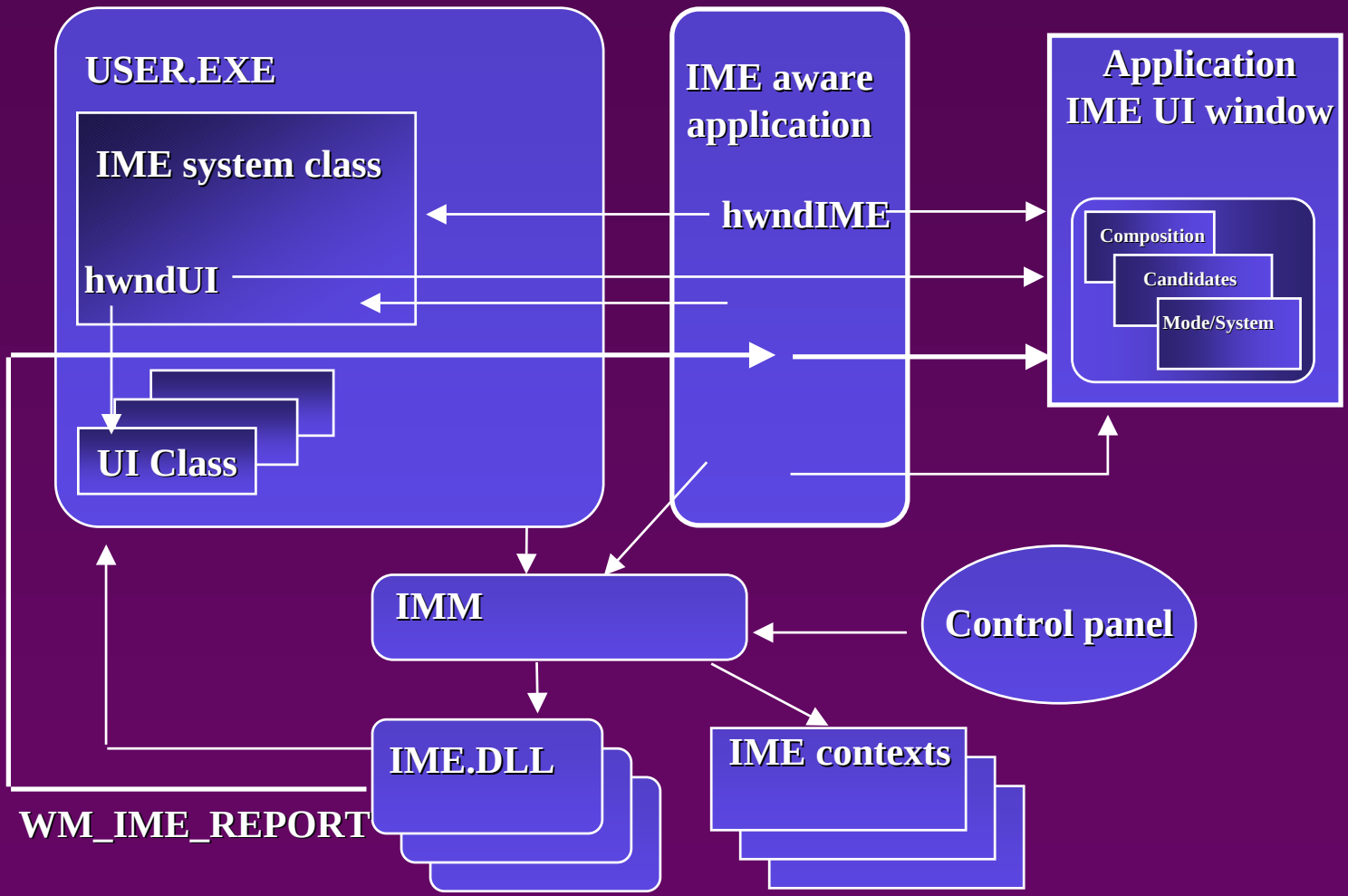




IME-Unaware Application

```
case WM_CHAR:
        if (!fWaitSecond )
        {
                szString[0]=wParam;
                szString[1]=0;
                if(IsDBCSLeadByte(wParam))
                {
                        fWaitSecond = TRUE;
                break;
                }
           }
           else
           {
                szString[1]=wParam;
                szString[2]=0;
                fWaitSecond = FALSE;
           }
         TextOut(hdc, x, y, szString, strlen(szString));
        break;
```

IME-Aware Applications



IME-Aware Application
Use IR_STRING

```
case WM_IME_REPORT:
        {
                if(wParam == IR_STRING)
                {

                        if (lpP = GlobalLock((HANDLE)LOWORD(lParam)))
                {
                                TextOut(hdc, x, y, lpP, lstrlen(lpP));
                GlobalUnlock((HANDLE)LOWORD(lParam));
                 return 1L; // processed
                }
        }
        }
           break;
```

System IME Class

- Predefined class like EDIT or LISTBOX class
- Carries out all IME user interface functions
- Does not receive direct user input but responds to IME messages
- Each application can have its own IME window and maintain state at task switch:
  - Incomplete input before conversion
  - Candidates list
- IME window will have application's handle
  - IME can trace caret position
  - IME move with application when moved

How To Use The IME Class

- Application creates its own IME class
  - With WS_DISABLE, without WS_VISABLE
  - Application's IME class will never handle input
- Application's IME class will create instance of currently selected IME user-interface window
- All IME message can be passed to the application's IME class

Is IME Available?
Conditional use of IME

```
struct _tagIMEfuncs  {
        LPCHAR   pszName;
        FARPROC *pFunc;
} IMEFunc[ NUMIMEFUNCS] = {
        "IMPAddIMEA", NULL, "IMPDeletIMEA", NULL, "IMPGetIMEA", NULL};


...
        if ( (hDLL = LoadIMEFuncs( IMEFunc, NUMIMEFUNCS)) )  {
                //... Do Japanese stuff
                ...
                call FreeLibrary( hDLL);
        }
...

HMODULE LoadIMEFuncs( struct _tagIMEFuncs * pIME, int cFuncs)  {
        HMODULE hDLL = LoadLibrary( TEXT("user32"));
        if ( hDLL )
          while ( cFuncs-- >= 0 )
            pIME[cFuncs].pFunc = GetProcAddress( hDLL, pIME[cFuncs].pszName);
        return( hDLL);
}
```

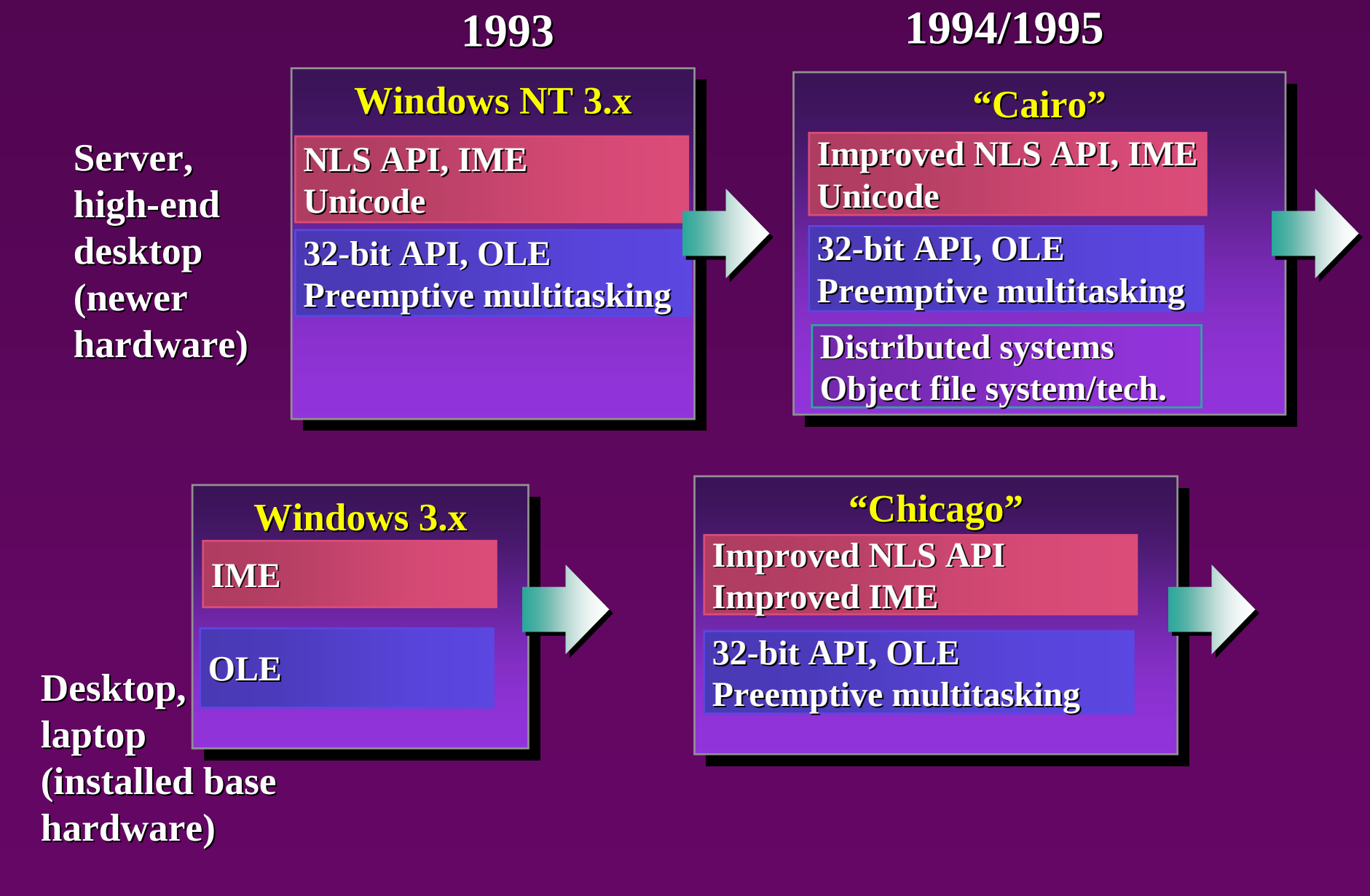
Agenda

- “Chicago” international product
- Win32 API for international
- Win32 input method editors
- **What to do**

International Win32

- Write to the Win32 API wherever possible
- Use the NLS APIs
- Use Windows NT or “Chicago” as dev platform
- Integrate OLE 2.0 now
- Consider direct Windows IME support

Evolution Of Windows



International Win32 Results: U.S. ISVs

32-bit Japanese public demos on Windows NT

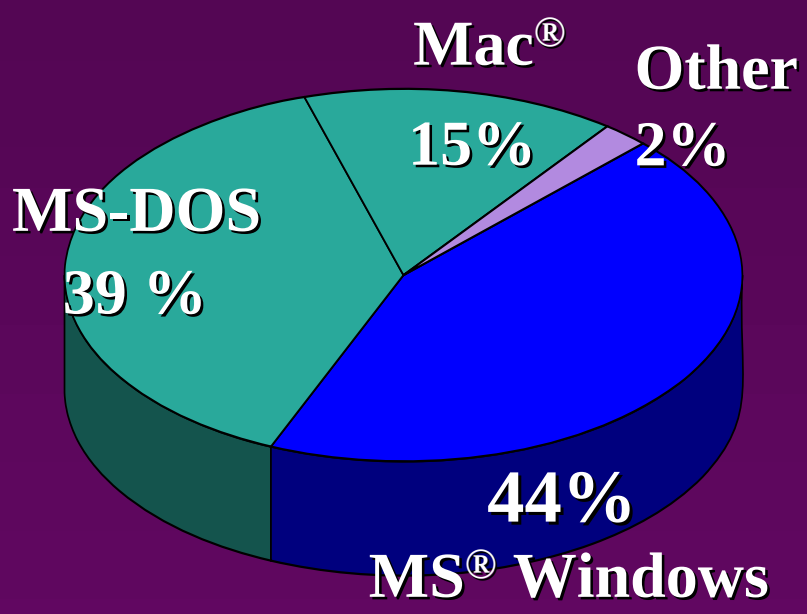
- **CAD/CAM:** Adra, Computervision, Parametric
- **Communications and network:** Eicon, NetManage
- **Database:** Microsoft, ORACLE
- **EIS:** Information Resources, SAS
- **Financial:** IMRS
- **Machine translation:** Language Engineering Corp
- **Manufacturing/Scientific:** National Instruments
- **Publishing:** Bitstream, Interleaf
- **Other applications not yet publicly demoed**

Things To Get

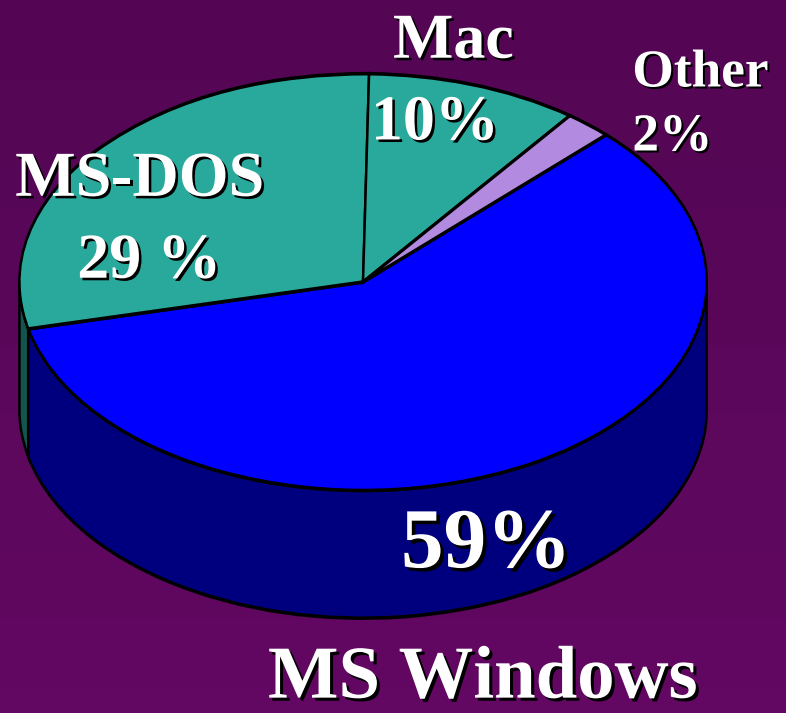
- All shipped international Windows, Win32, Win32s, SDKs, DDKs, all in one place: MSDN Level 2, (800) 759-5474
- Compiler with libraries for ANSI, DBCS, and/or Unicode
- Book: *"International Handbook for Software Design,"* on MSDN Level 1
- RLMan, RLTools on MSDN Level 1
- Globalization Resource Kit: guide to Windows localizers, international consultants, book list, etc. from global@microsoft.com
- UI terminology for Europe: The GUI Guide International from (800)MS-PRESS
- UI terminology for Far East in SDK glossaries

Why It Matters

Applications revenue by platform



USA



International

Source: SPA Jan.-Jun. ’93

Summary

- Technical barriers to places like the Far East dramatically reduced during the past year
- Win32 API unlocks all future versions of Windows and is available today
- Win32 is not just for the U.S.
- OLE technology is core to “Chicago”, and even more so with “Cairo”
- Don’t just run on Windows, exploit its newest technologies to your maximum advantage

Technical Strategy

- Port your existing applications to Win32
- Follow the “great application” guidelines Robert Hess presentation tomorrow
- Mail winbeta@microsoft.com find out how to become a pre-release candidate
- Test your applications across the Windows family