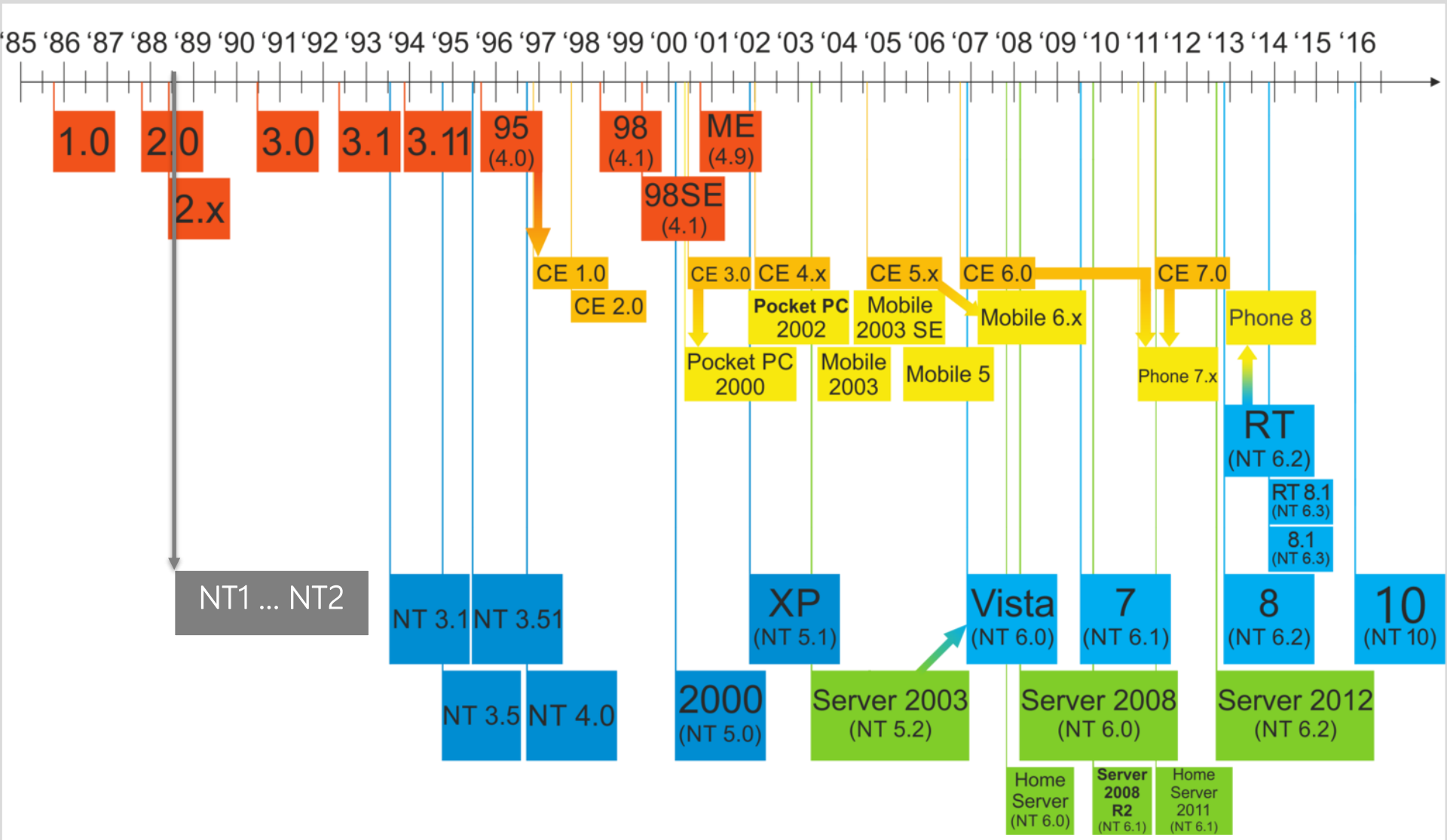
Cours Windows IoT

1 – Introduction Win32

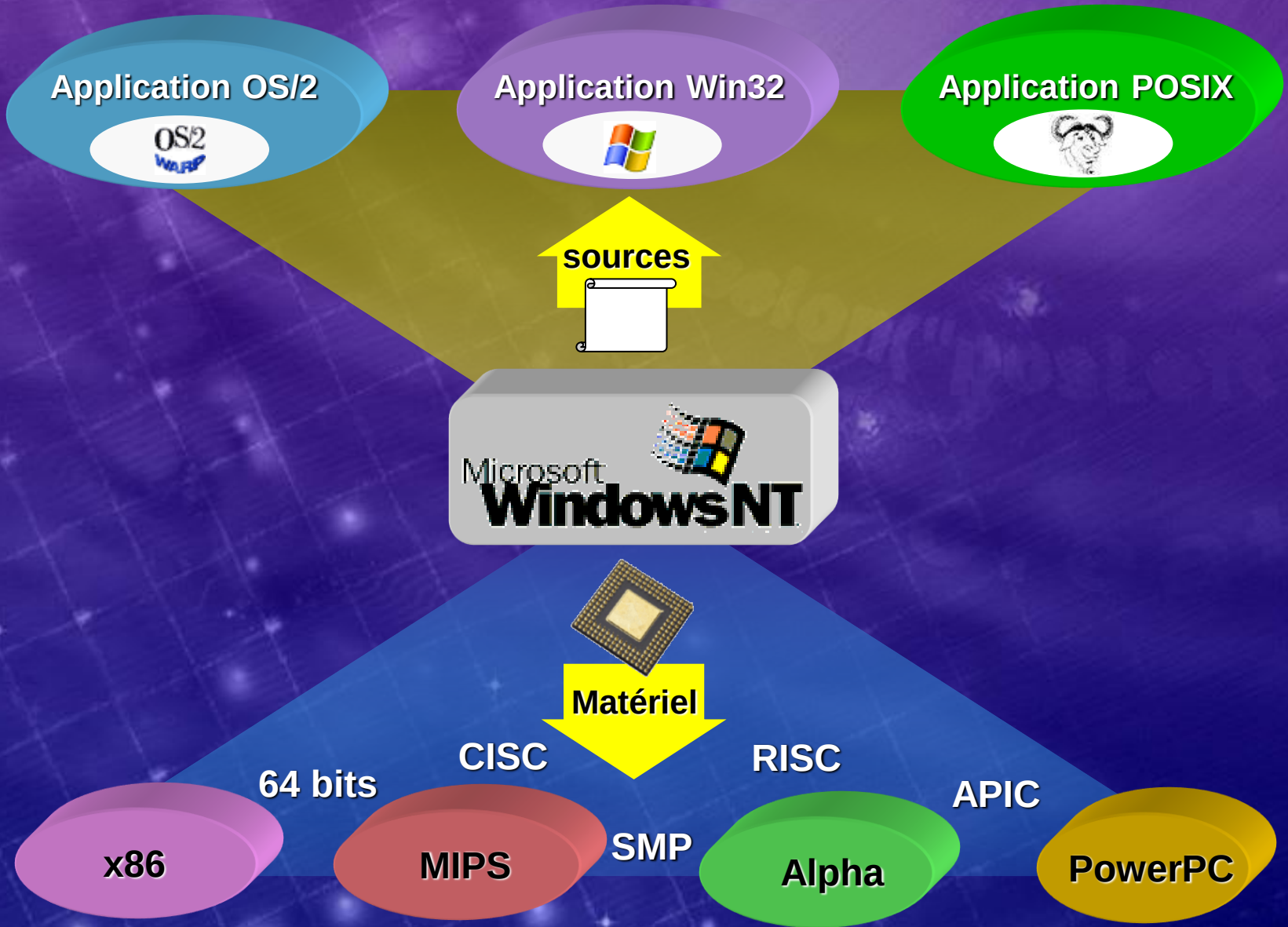
Plan du chapitre

- Historique Windows
- Les systèmes d'exploitation Microsoft
- L'architecture Windows XP
- Des outils
- L’API Win32

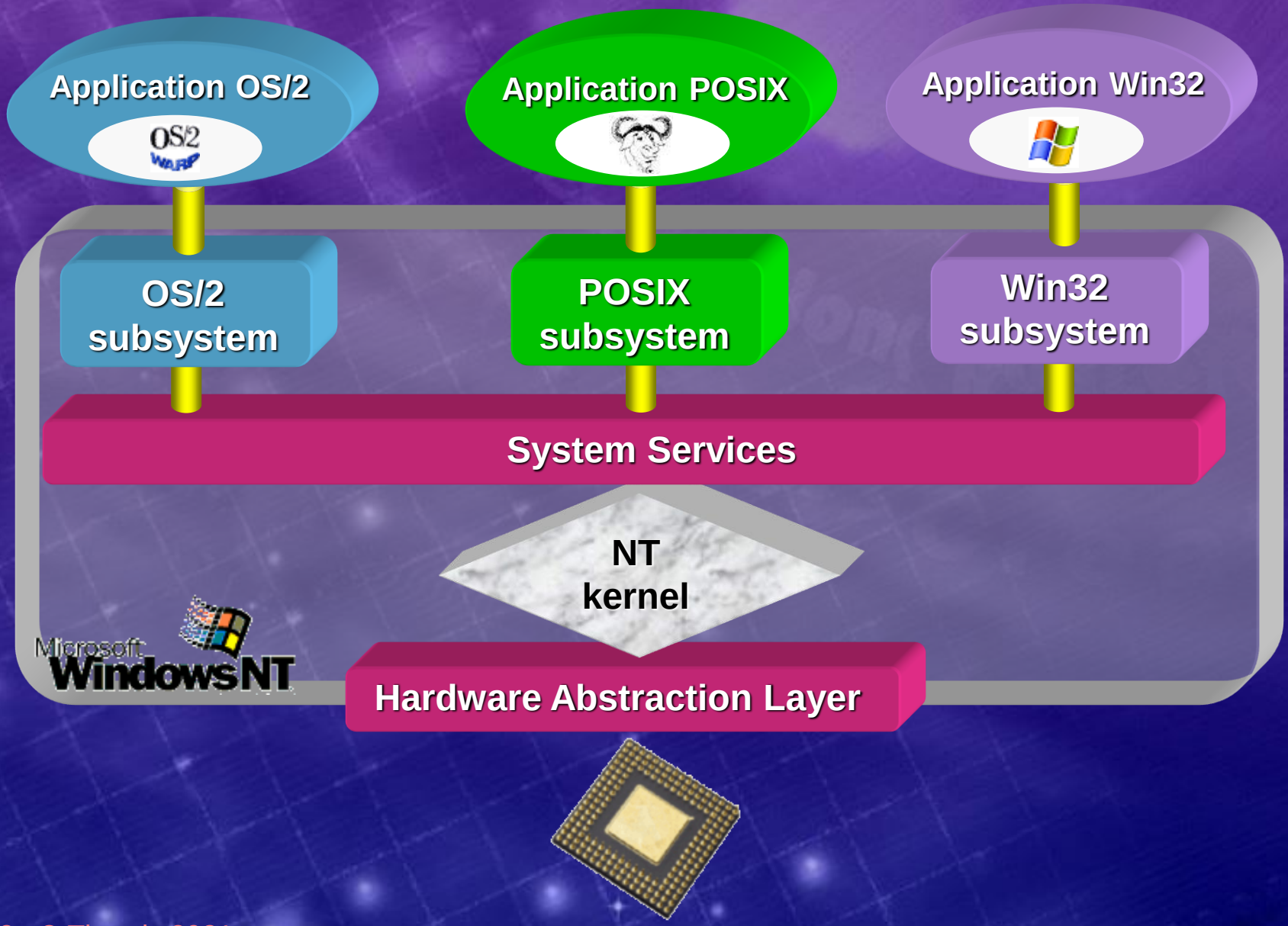
Chronologie des Windows



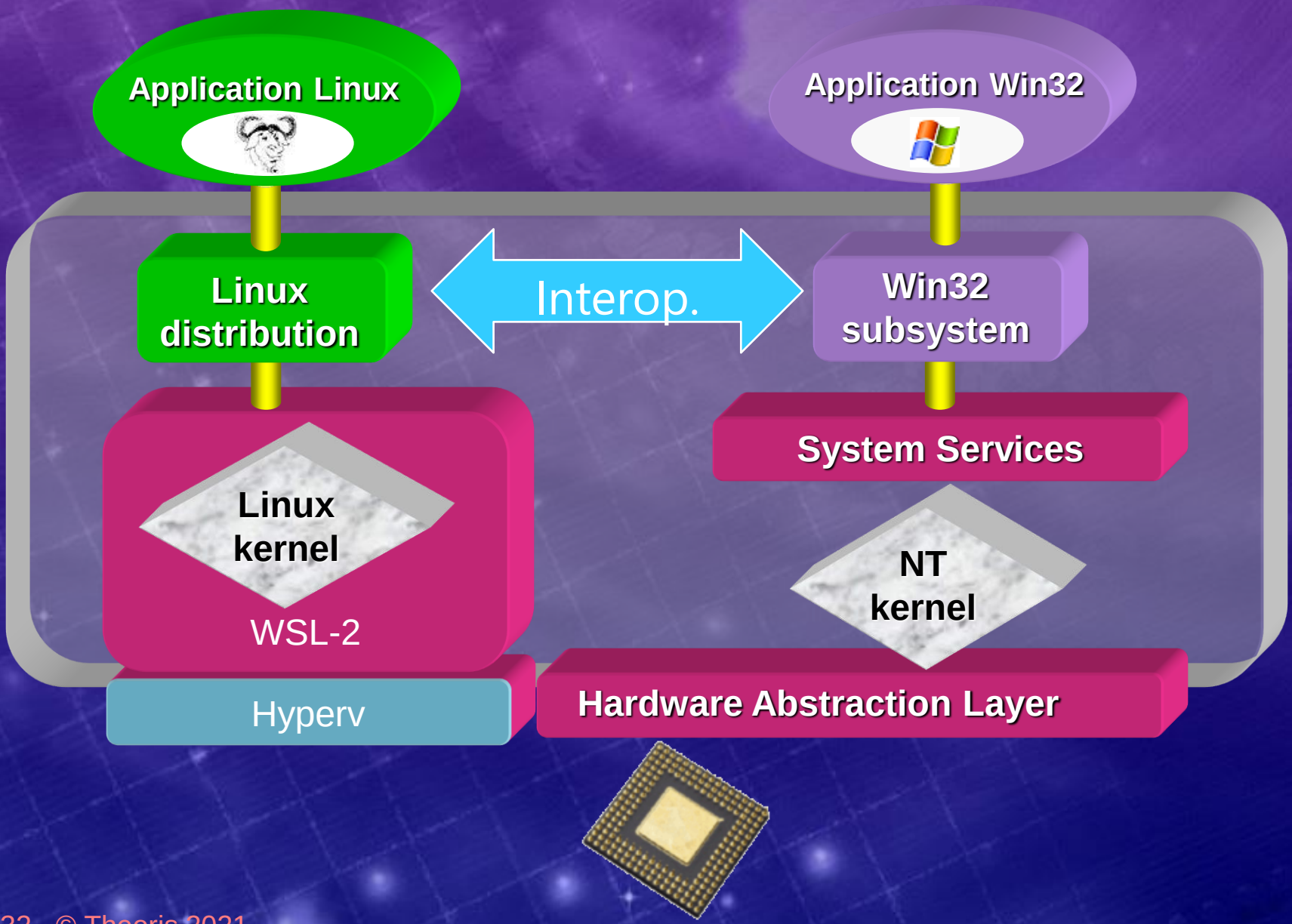
Objectifs de NT



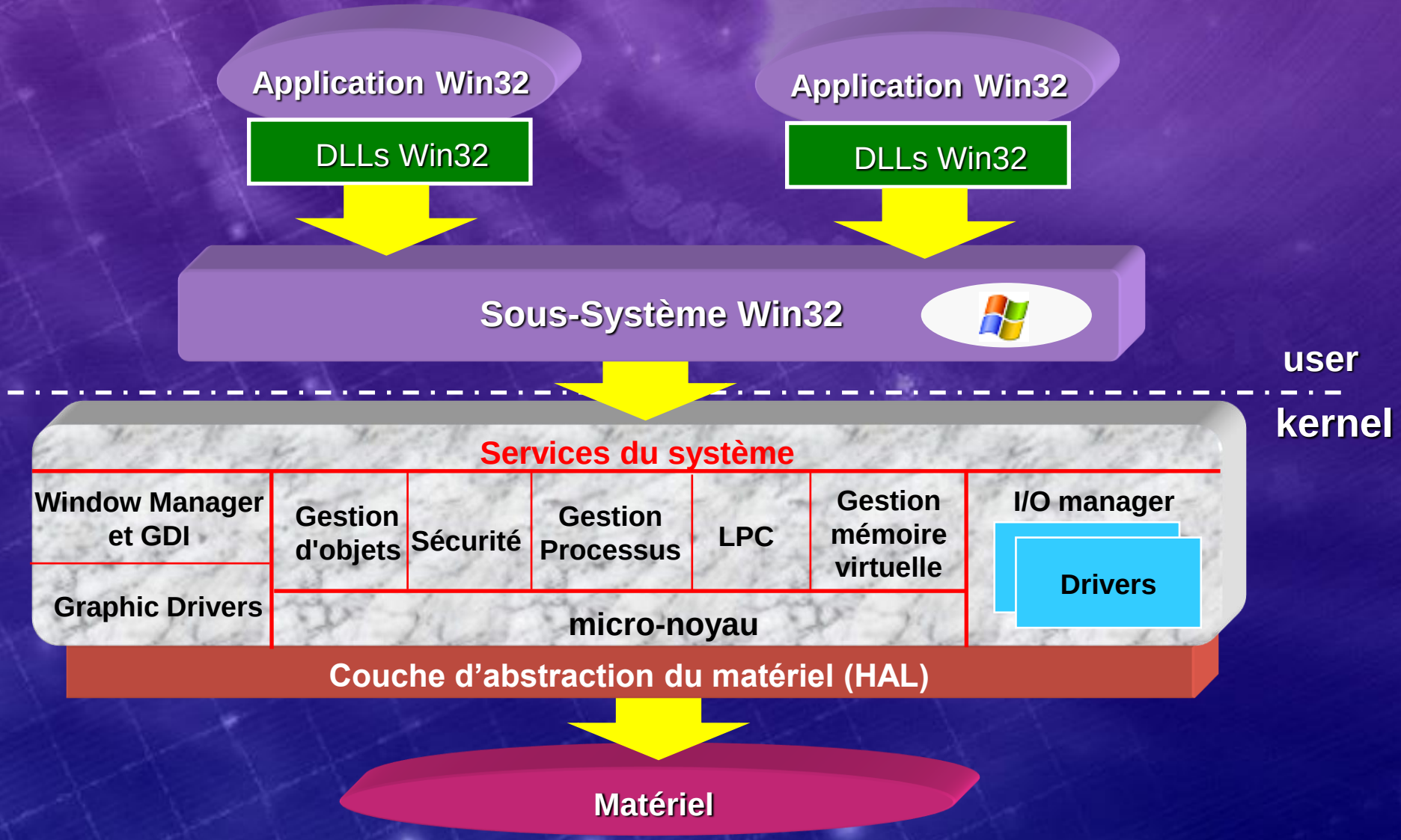
Choix d'architecture NT



2021 – WSL-2


Application Linux
Application Win32
Linux distribution
Interop.
Win32 subsystem
System Services
Linux kernel
WSL-2
NT kernel
Hyperv
Hardware Abstraction Layer

Kernel NT


Application Win32
DLLs Win32
Application Win32
DLLs Win32
Sous-Système Win32
user
kernel
Services du système
Window Manager et GDI
Graphic Drivers
Gestion d'objets
Sécurité
Gestion Processus
LPC
Gestion mémoire virtuelle
micro-noyau
I/O manager
Drivers
Couche d'abstraction du matériel (HAL)
Matériel

Mode “Kernel” & Mode “User”

- Deux modes gérés par Windows XP
  - Kernel (OS)
    - Mode CPU privilégié
    - Accès aux données du système et au Hardware
  - User (Applications)
    - Mode CPU non-privilégié
    - Pas d’accès au hardware et aux données système
- Les applications sont séparées de l’OS
  - Ells ne peuvent pas modifier son intégrité
- Les composants système et les pilotes peuvent atteindre les donnée du système

UTILISER NT

Raccourcis clavier

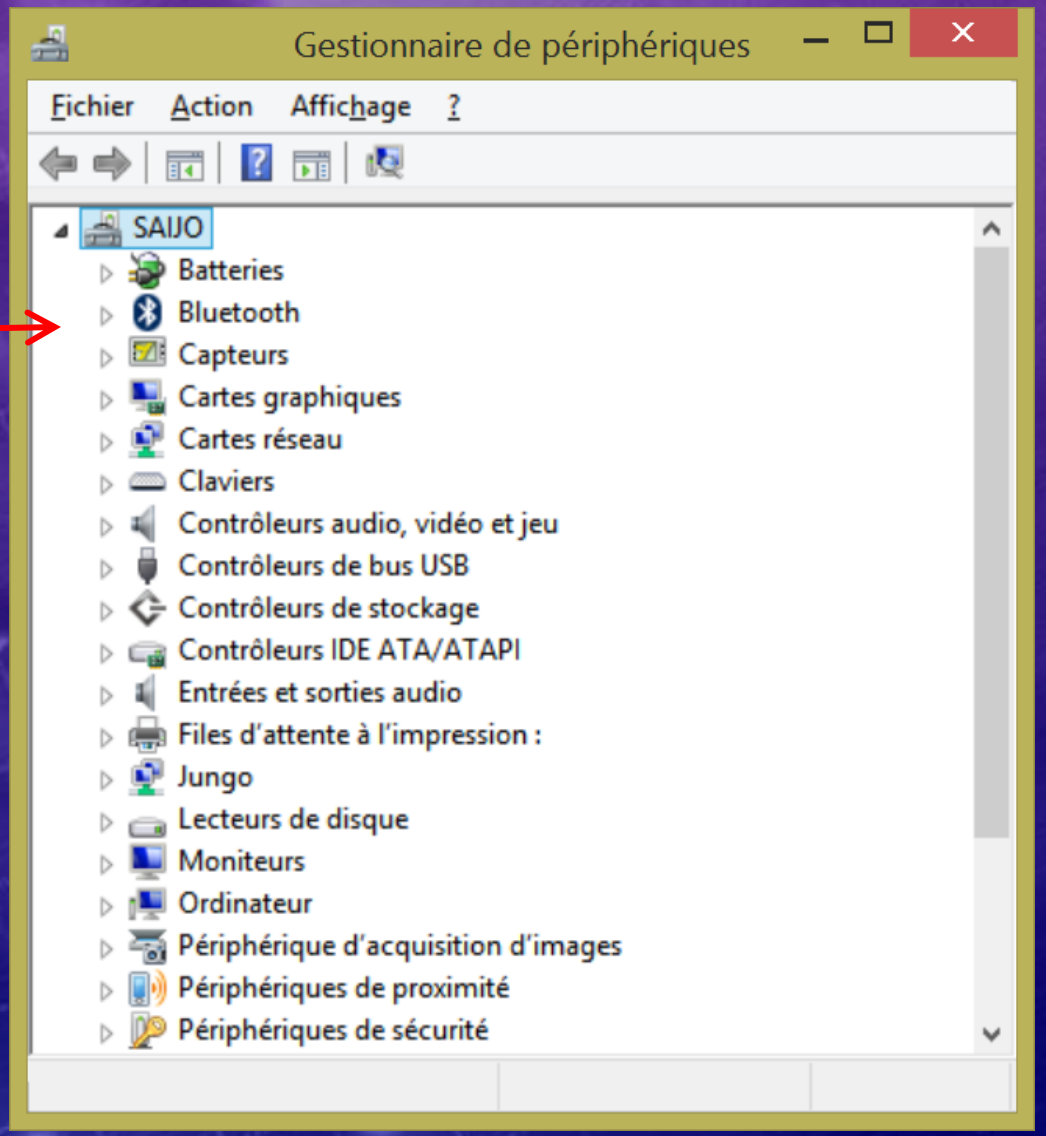
- ⌘ + ‘r’ → run
- ⌘ + ‘e’ → explorer
- ⌘ + ‘p’ → affichages
- ⌘ + ‘l’ → lock session
- ⌘ + ‘d’ → basculer sur le bureau
- Alt + ‘impr ecran » → fenetre courante
- Alt + ‘tab’ → bascule applications
- Etc.

Les outils Windows

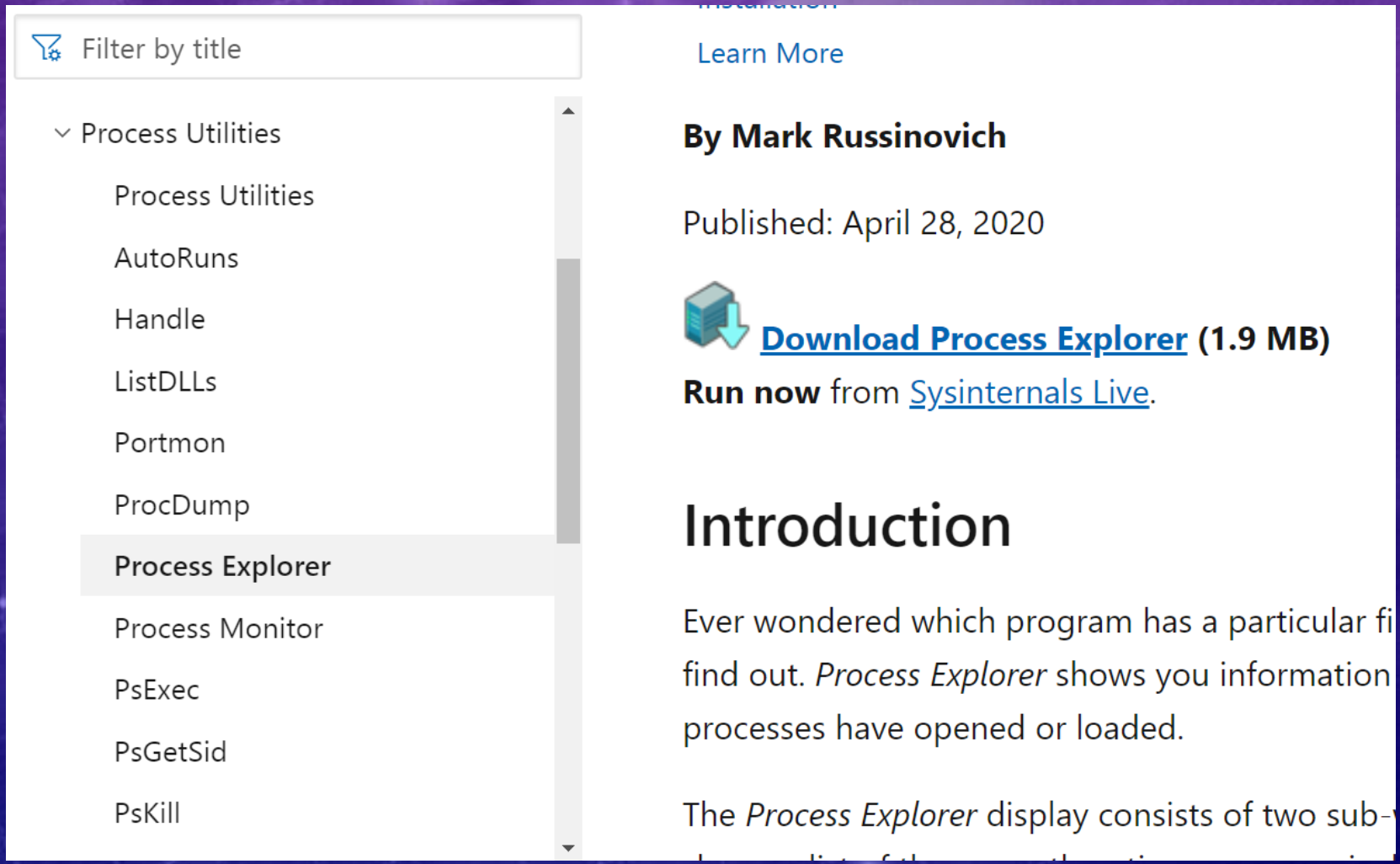
- Terminal windows
  - Cmd.exe
- Outils dans `%systemroot%\system32`
  - msinfo32.exe
  - Perfmon.exe
  - Taskmgr.exe
  - Regedt32.exe, regedit.exe
  - Ping.exe, tracert.exe
  - Etc.. Etc. etc.

Les consoles Windows

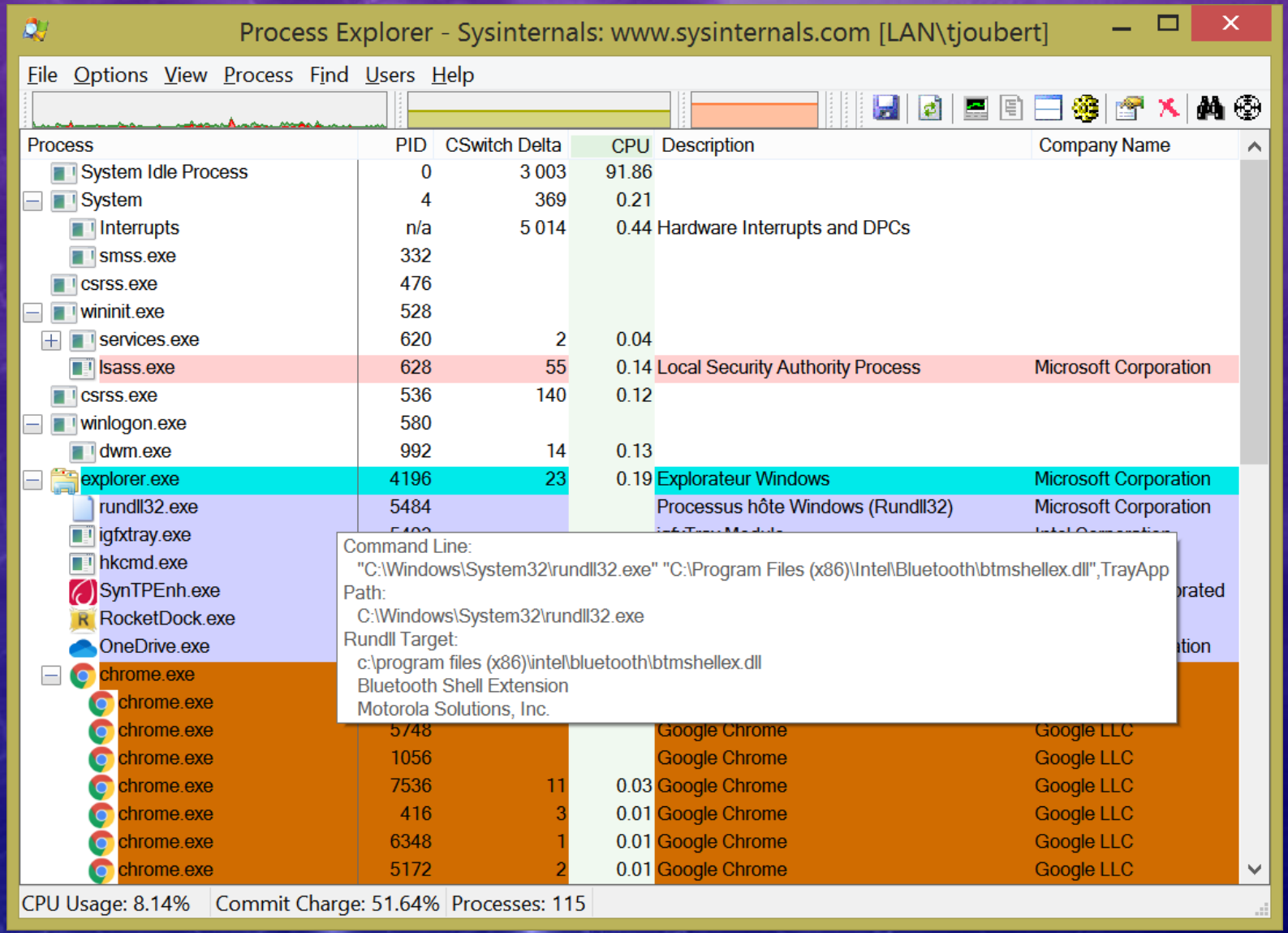
- azman.msc
- certlm.msc
- certmgr.msc
- comexp.msc
- compmgmt.msc
- devmgmt.msc
- diskmgmt.msc
- eventvwr.msc
- fsmgmt.msc
- gpedit.msc
- lusrmgr.msc
- NAPCLCFG.MSC
- perfmon.msc
- printmanagement.msc
- rsop.msc
- secpol.msc
- services.msc
- taskschd.msc
- tpm.msc
- WF.msc
- WmiMgmt.msc



Les outils sysinternals

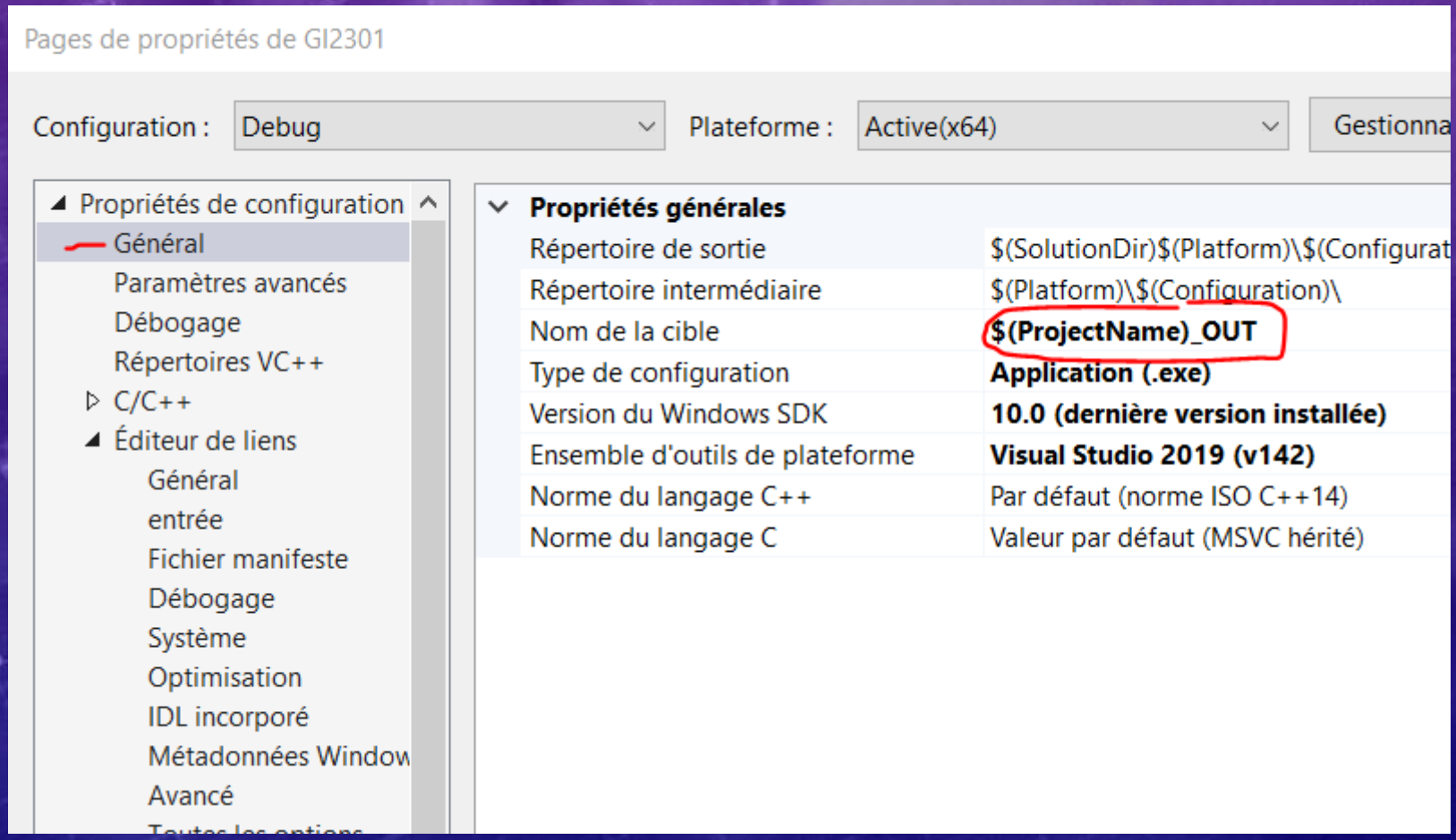


L'outil procexp.exe

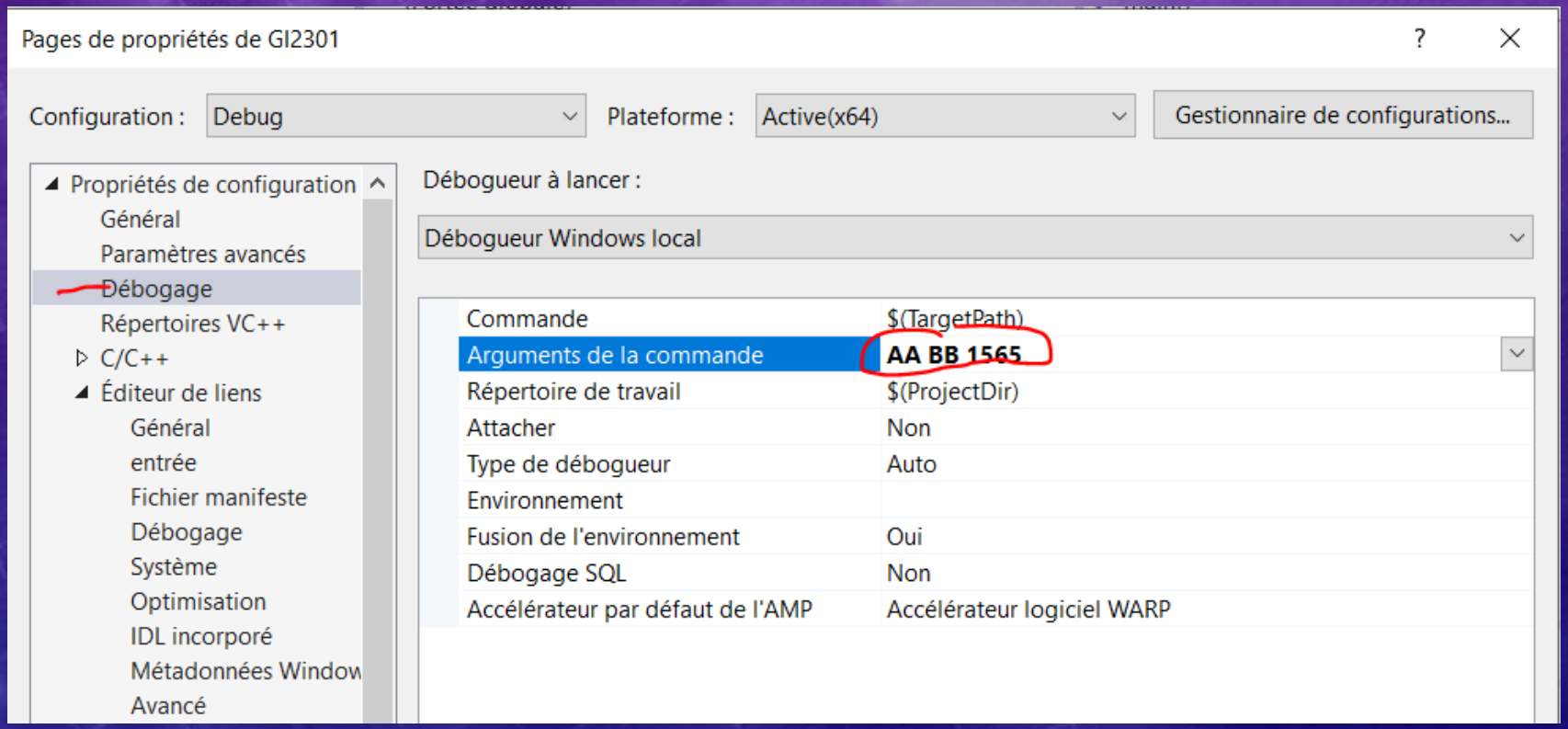


VISUAL STUDIO ESSENTIALS

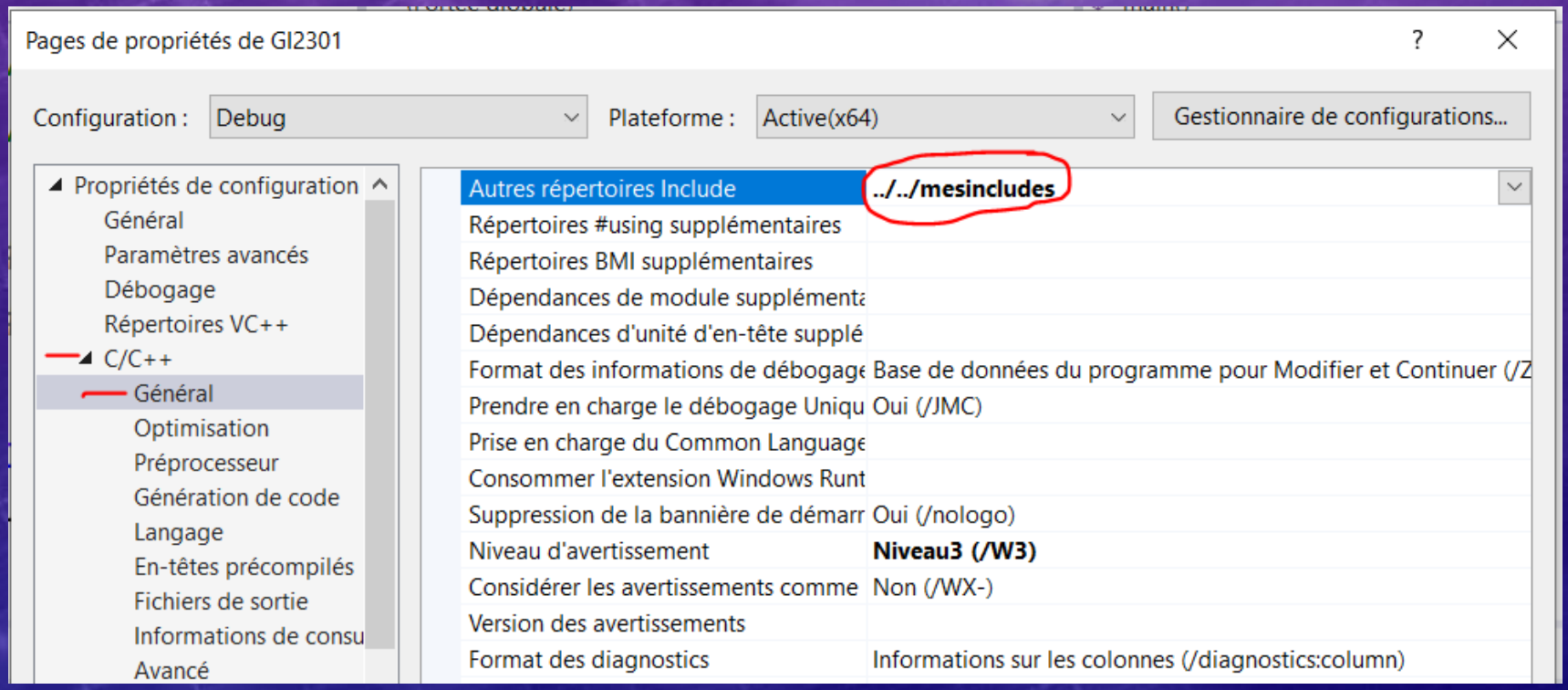
VS output file



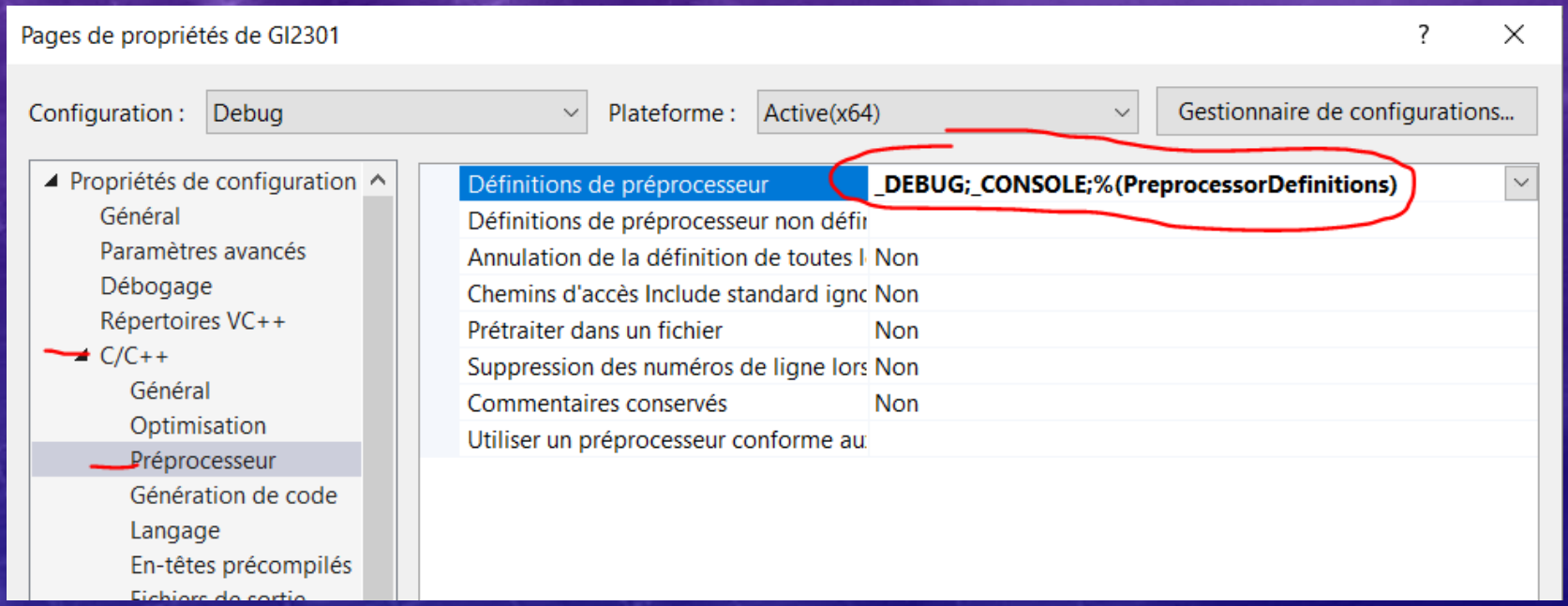
VS arguments au lancement



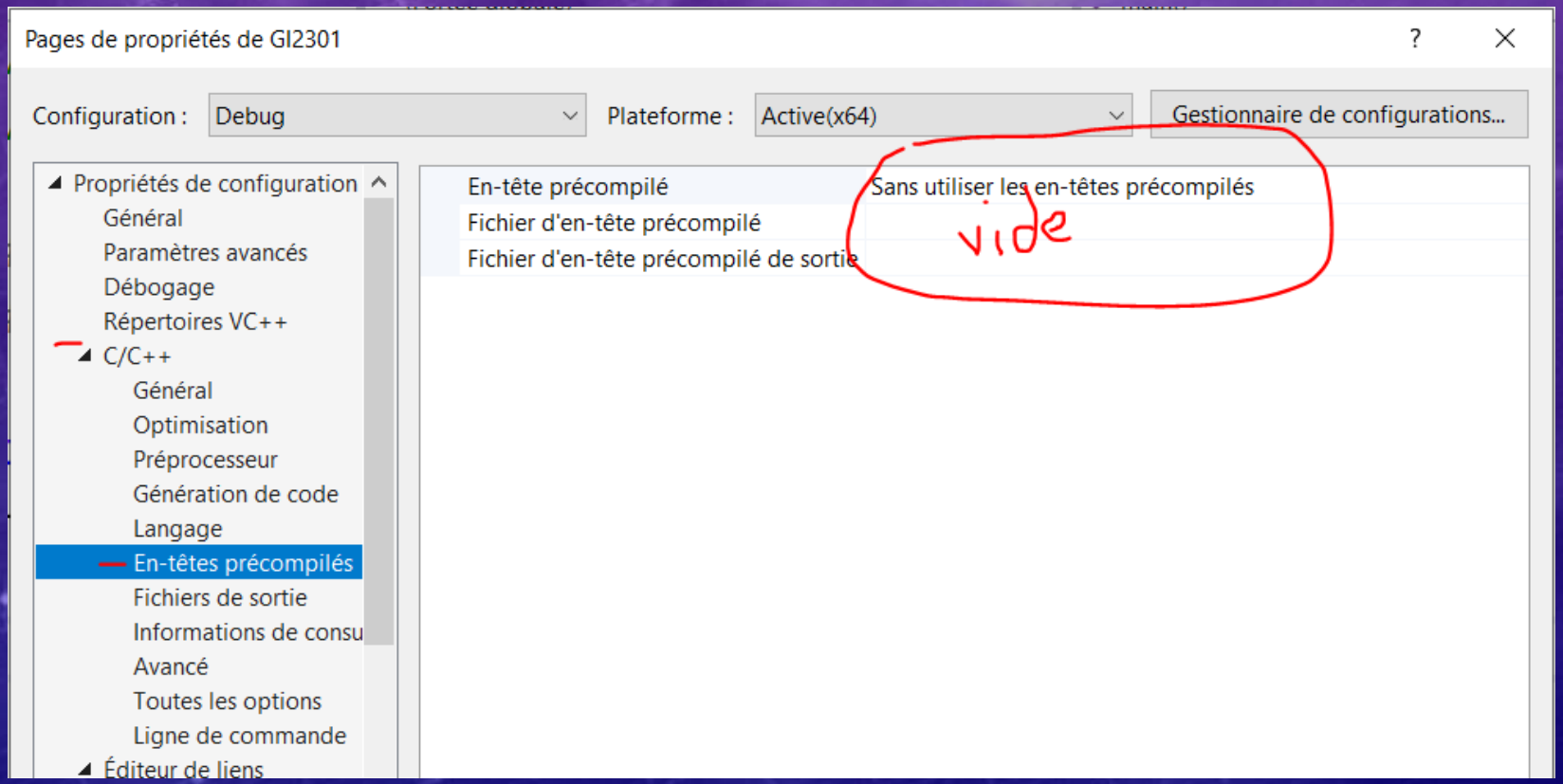
VS chemins d'includes



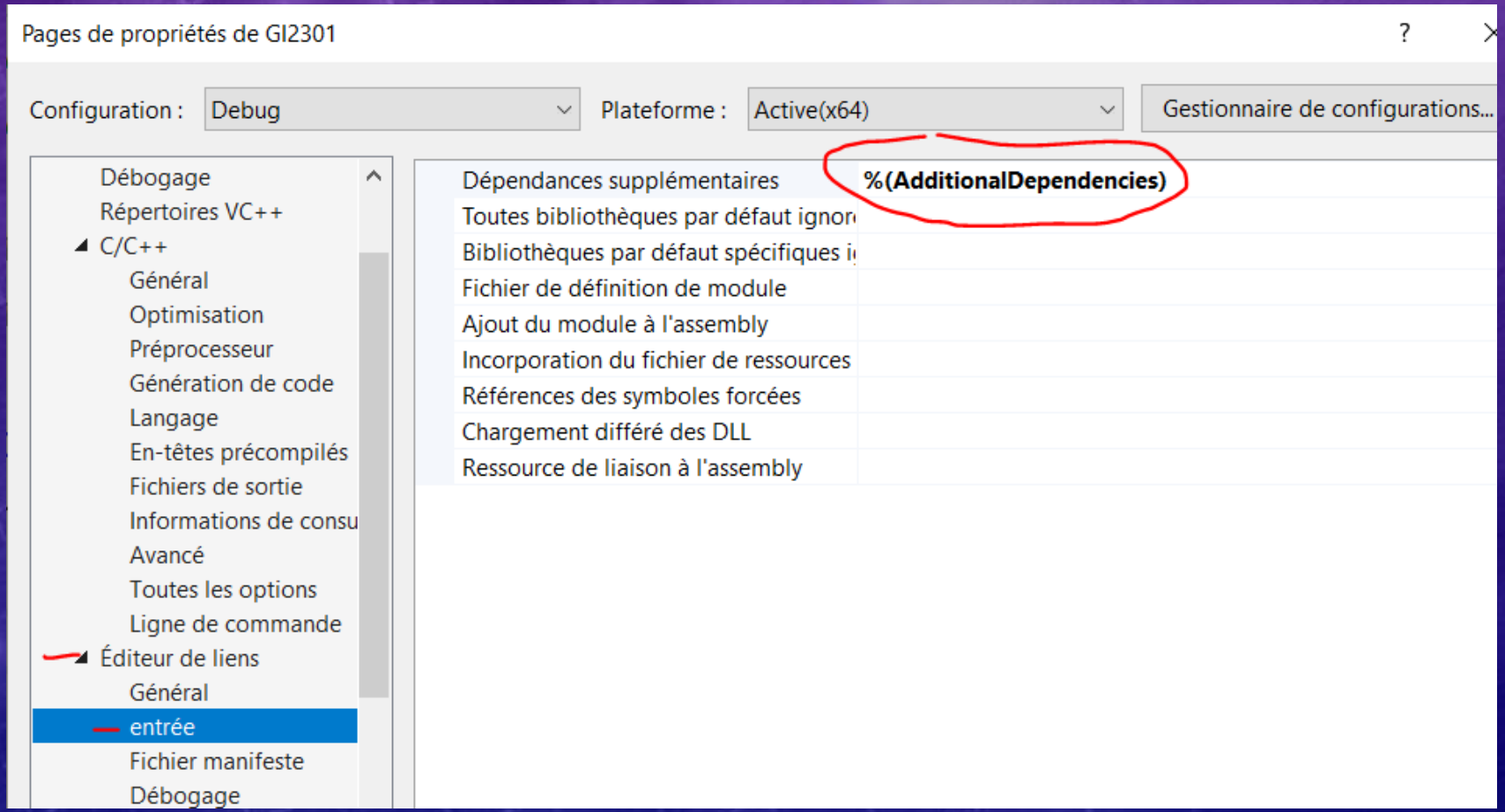
VS macros de compilation



VS no .PCH


Pages de propriétés de GI2301
Configuration : Debug
Plateforme : Active(x64)
Gestionnaire de configurations...
Propriétés de configuration
Général
Paramètres avancés
Débogage
Répertoires VC++
C/C++
Général
Optimisation
Préprocesseur
Génération de code
Langage
En-têtes précompilés
Fichiers de sortie
Informations de consu
Avancé
Toutes les options
Ligne de commande
Éditeur de liens
En-tête précompilé
Sans utiliser les en-têtes précompilés
Fichier d'en-tête précompilé
Fichier d'en-tête précompilé de sortie
vide

VS librairies



API WIN32 ESSENTIALS

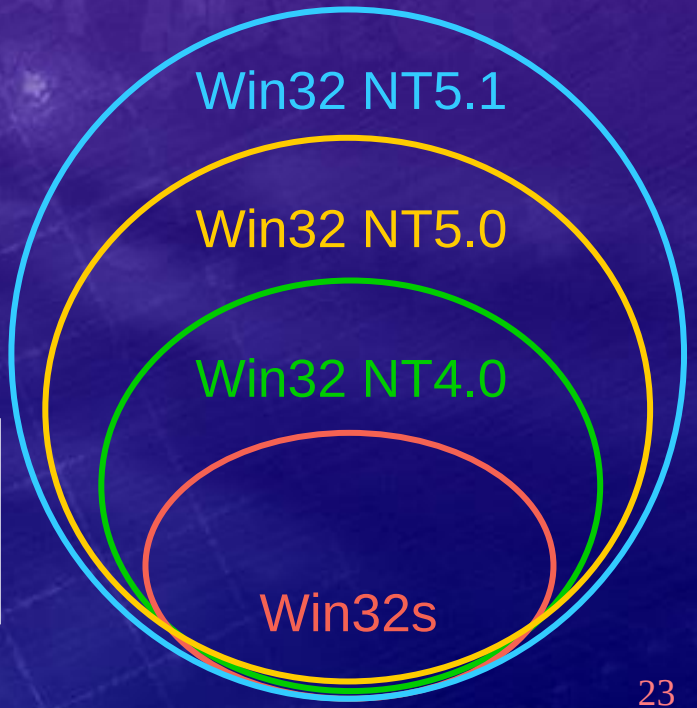
L'API Win32

- `#include <windows.h>`

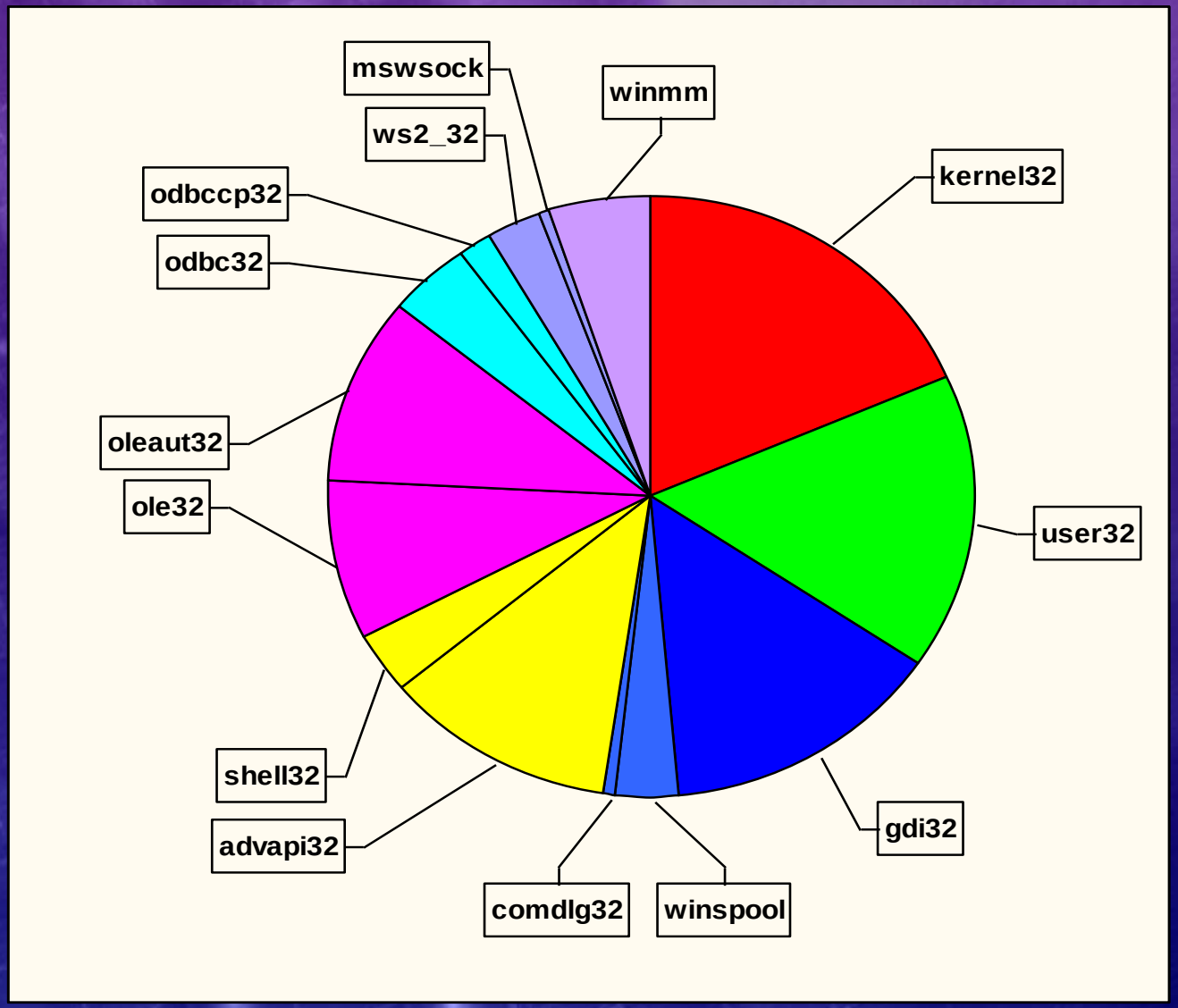
- API en C , des milliers de fonctions, de messages de types et de constantes

- Win32s, Win32c, etc.
  - Née avec Windows NT
  - Win32s sous Windows 3.11
  - Des évolutions à chaque version
  - Périmètre de compilation:

```
#define  _WIN32_WINNT  0x0510
#include « windows.h »
```



Contenu de l'API Win32



Unicode & langage C

- Type compatible Unicode

```
wchar_t
```

- Fonctions compatibles Unicode

```
wcs…, _wcs…
```

- Chaines constantes

```
L"je suis Unicode"
"je suis ASCII"
```

- Fonctions génériques

```
#define _UNICODE
#include <tchar.h>
#include <wchar.h>

_tcs…
```

Unicode & Windows

- Types génériques

```
#ifdef UNICODE
        typedef wchar_t TCHAR;
#else
        typedef unsigned char TCHAR
#endif
```

→ TCHAR, LPTSTR, LPTCH

- Taille des chaînes

```
cByteCount = (lpEnd-lpStart) * sizeof(TCHAR);
```

- La macro TEXT( ) pour les constantes

```
while(*lpFileName++ != TEXT( '\\' ) )
```

Unicode & API win32

- Version générique

```
BOOL SetWindowText( HWND hwnd, LPCTSTR lpText );
```

- Version ANSI

```
BOOL SetWindowTextA(HWND hwnd, LPCSTR lpText );
```

- Version UNICODE

```
BOOL SetWindowTextW(HWND hwnd, LPCWSTR lpText );
```

- Géré par le préprocesseur

```
#ifdef UNICODE
#define SetWindowText SetWindowTextW
#else
#define SetWindowText SetWindowTextA
#endif
```

Les types Windows

- API issue d'un monde 16bits, besoin de maîtriser les tailles → redéfinition des types du langage C
- Search « Windows Data types ».

DWORD	Mot de 32 bits non signé	Unsigned long
INT	Mot de 32 bits signé	int
LPVOID	Pointeur universel	Void*
LPSTR	Chaine « C »	char *
	Etc…	