Windows CE 6.0下的驱动程序新特性

何宗键
“微软——同济”移动与嵌入式中心
同济大学软件学院

功能推荐：

查询在线课程

搜索热门下载

讲座预约提醒

点击添加**MSN**机器人小新

为您收听下载**MSDN**中文网络广播课程加油助力**!**

aressoft@hotmail.com

Microsoft

本次课程内容包括

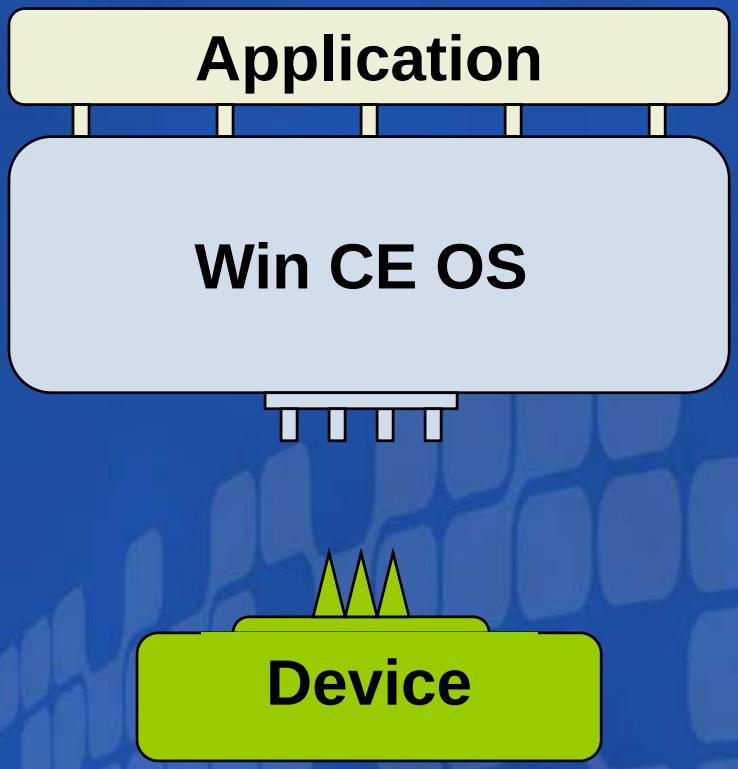
- 驱动程序开发简介
- CE 6.0的新内核架构对驱动程序的影响
- 内核模式驱动 VS 用户模式驱动
- 管理缓冲区
- 驱动程序的移植

收听本次课程需具备的条件

- 有过Windows CE 5.0开发经验
- 最好熟悉驱动程序开发
- 对CE 6内核有所了解

Level 300

驱动程序是什么？

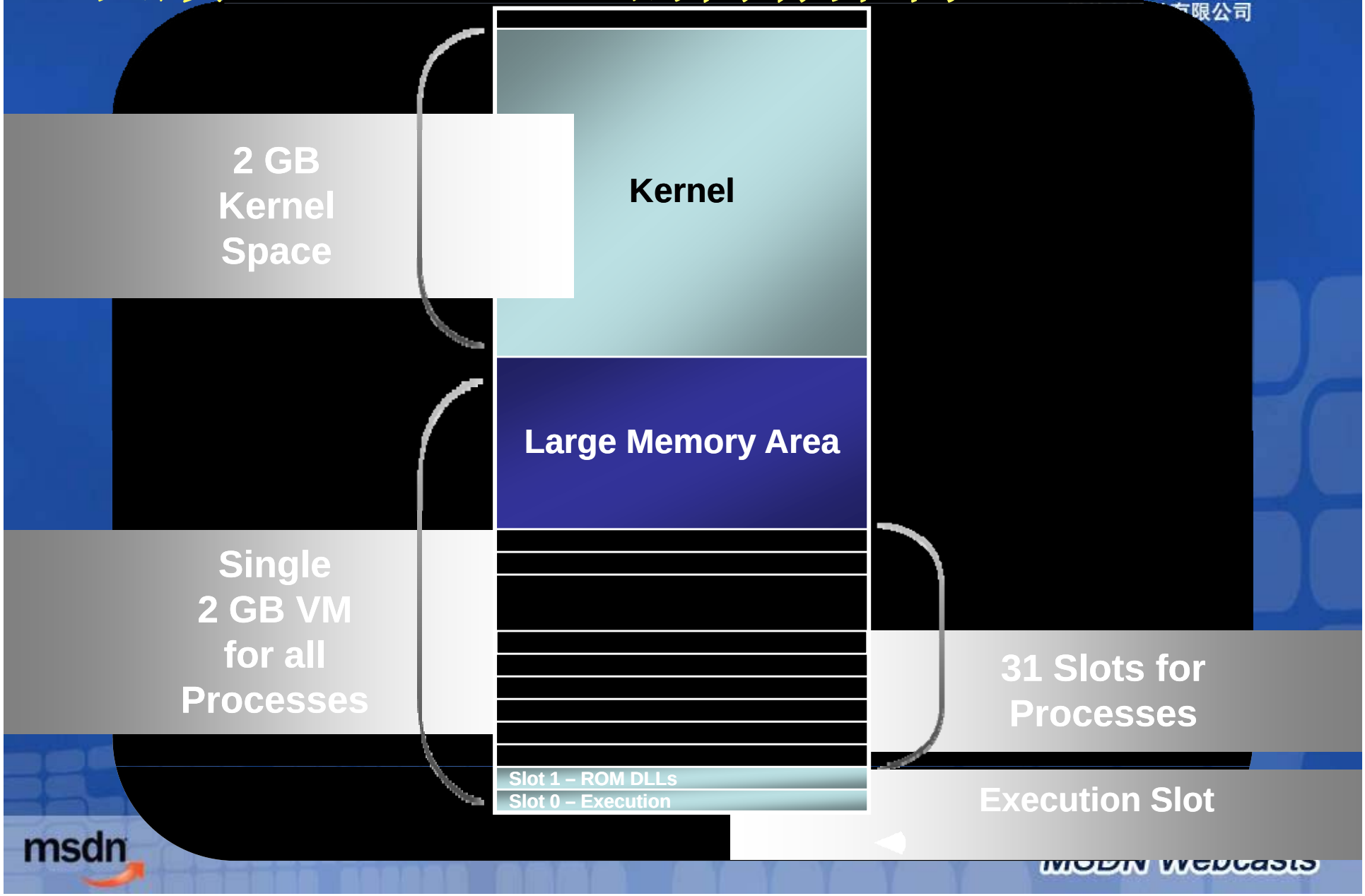


驱动程序是对物理或虚拟设备的抽象

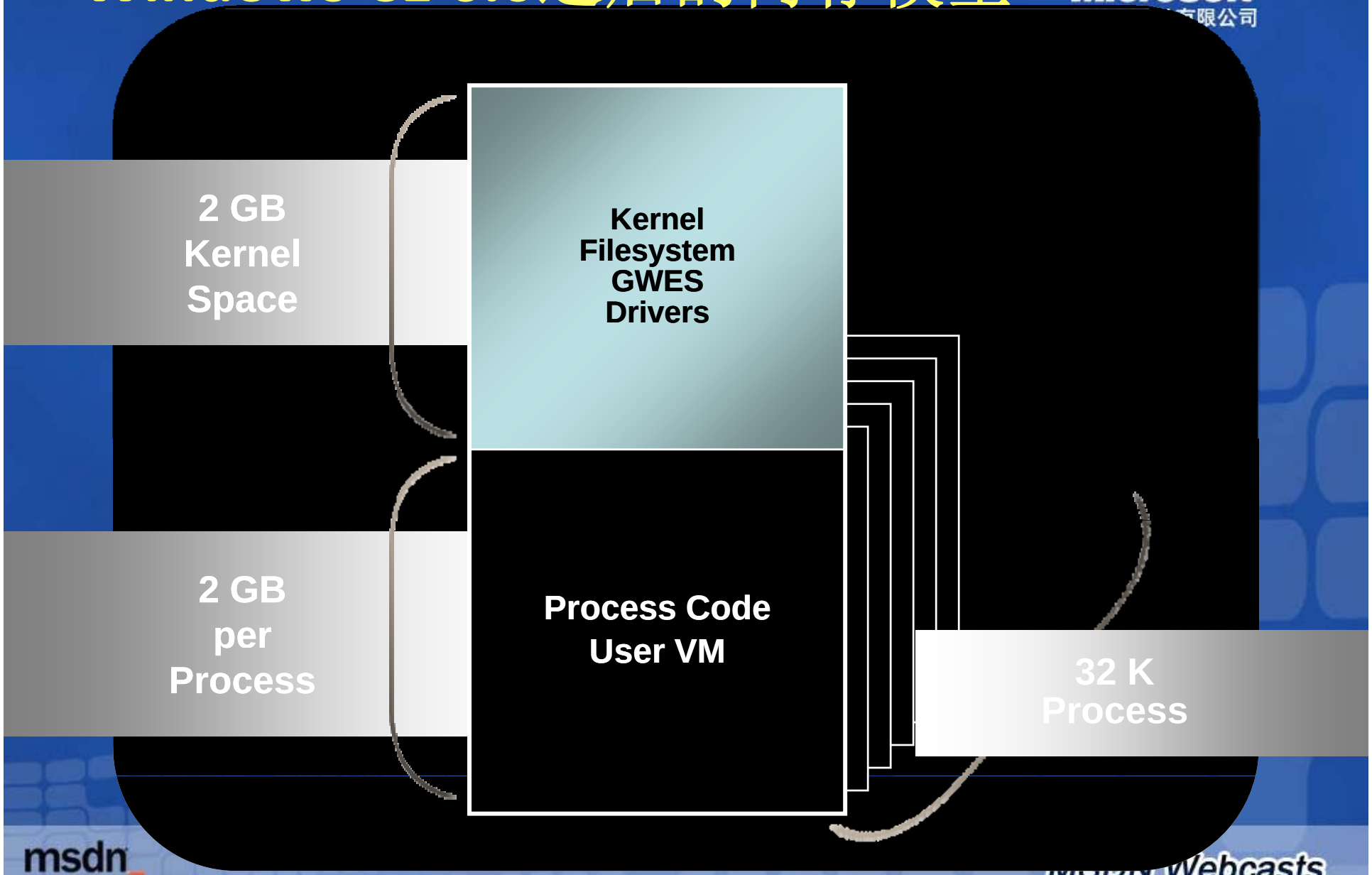
回顾驱动程序的基本知识

- Native vs Stream
  - 谁加载该驱动程序 (Who)
- 单体 vs 分层（MDD / PDD）
  - 驱动的结构如何设计 (How)
- Built In vs 动态加载
  - 驱动何时加载 (When)
- Miniport、Wave、Touch
  - 驱动的模型是什么样子 (What)
- 内核驱动 vs 用户驱动 （新）
  - 在哪里加载驱动 (Where)

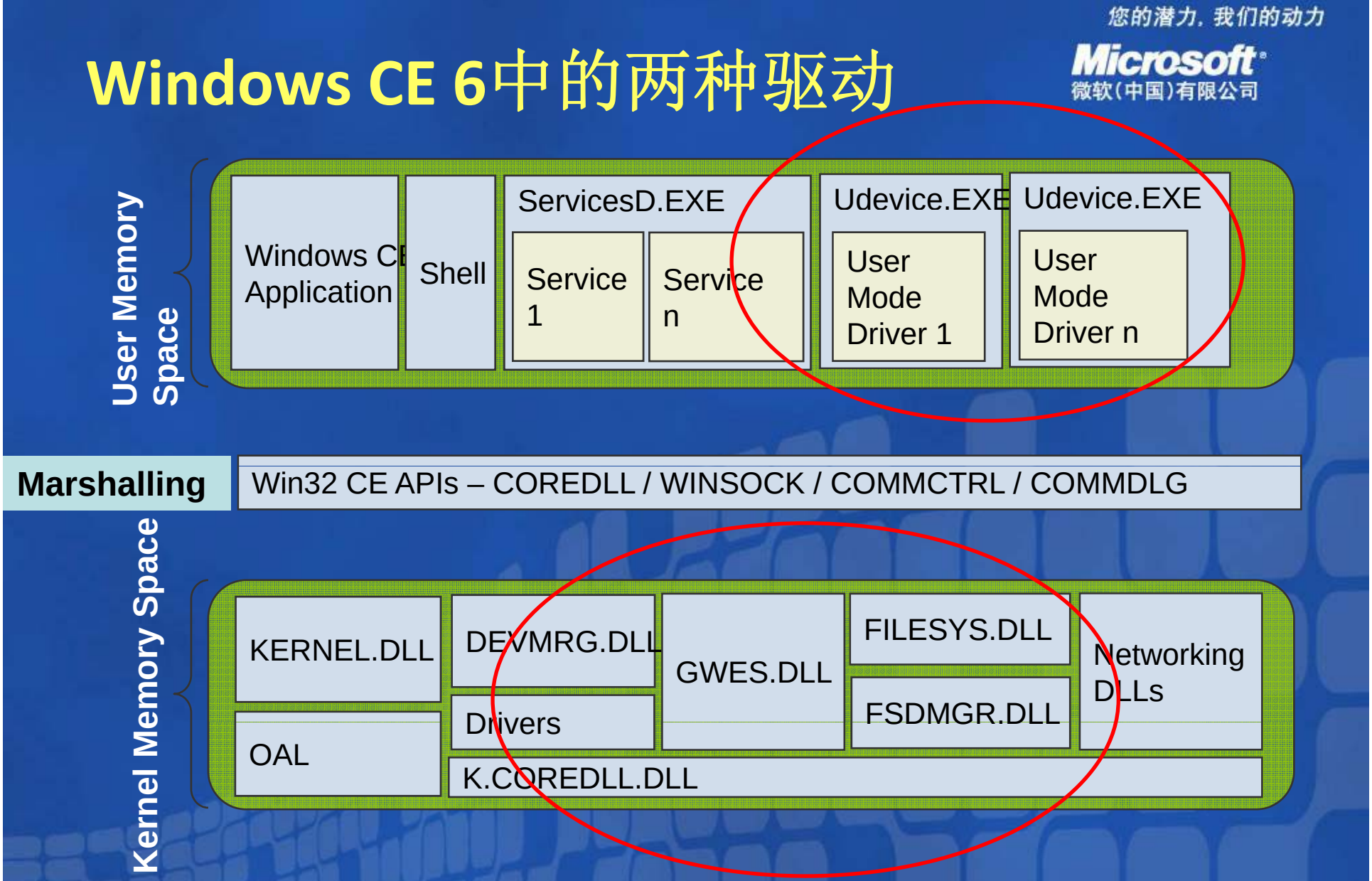
回顾Windows CE 5的内存架构



Windows CE 6.0之后的内存模型



Windows CE 6中的两种驱动



内核态驱动——新瓶装旧酒

- 默认的驱动模式
- 在内核地址空间中运行
- 与内核模式的coredll, k.coredll.dll链接
  - 自动完成，无需更改构建规则
- 效率高
  - 快速访问内核API
  - 直接同步访问用户的缓冲区
- 稳定压倒一切！
  - 驱动Crash可能会导致内核崩溃

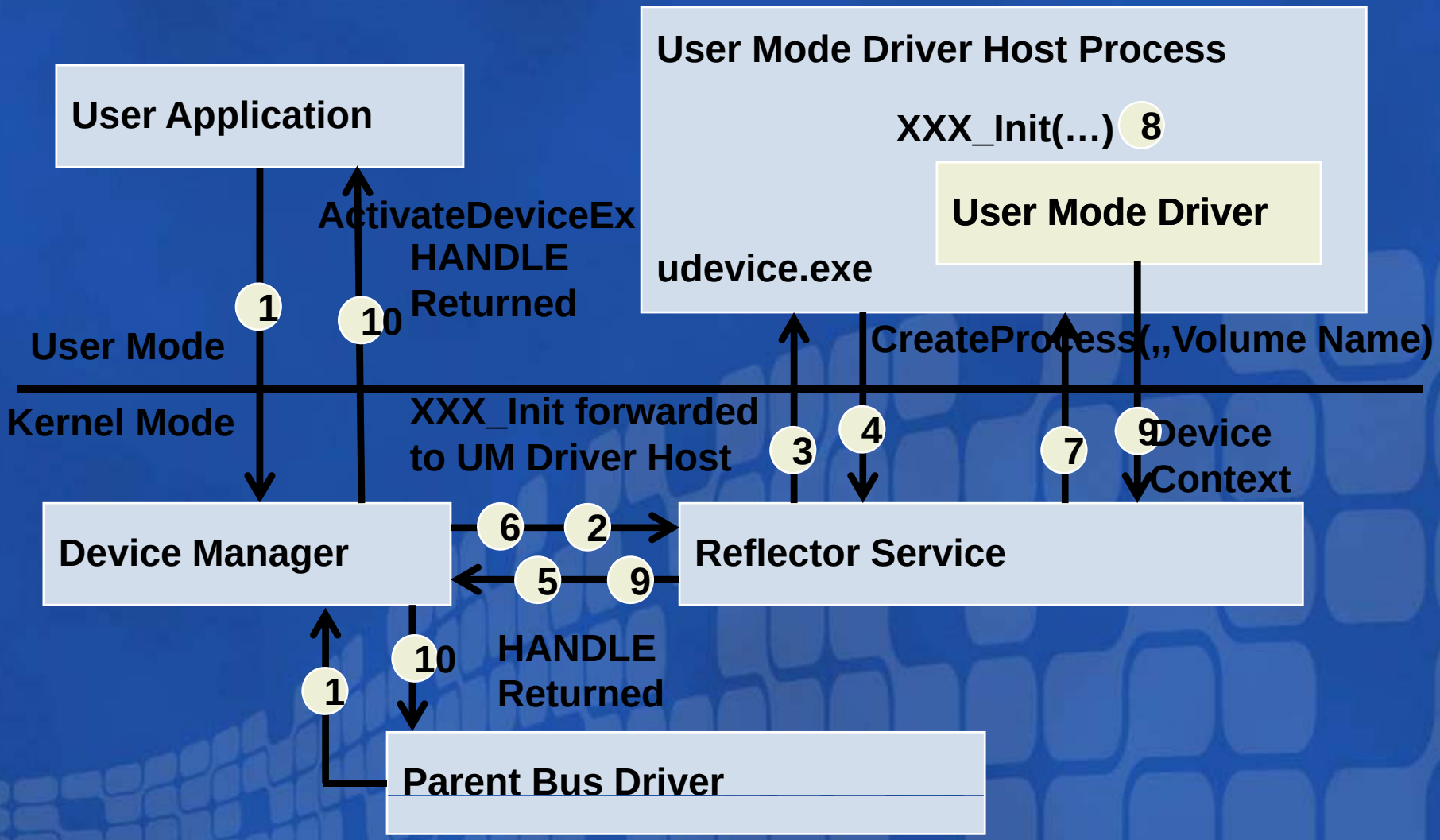
用户态驱动——老革命遇到新问题

- 注册表： DEVFLAGS_LOAD_AS_USERMODE
- 同样由设备管理器管理
- 被udevice.exe加载
- 与内核模式驱动基本兼容
- 用户模式驱动没有内核访问权限
  - 无法访问内核使用的虚拟内存
  - 无法调用部分内核专有API
- 可以增强系统的稳定性、驱动的灵活性
- 例子
  - 扩展总线设备，例如：USB、SDIO

Reflector服务

- 位于设备管理器中
- 每一个用户模式驱动对应一个Reflector
- 把设备的请求发送给用户态的驱动程序
- 代替用户模式驱动执行内核请求
- 让应用程序看起来两者之间没有区别

用户模式驱动程序加载



用户模式驱动注册表


```
[HKEY_LOCAL_MACHINE\Drivers\ProcGroup_0003]
   "ProcName"="udevice.exe"
   "ProcVolPrefix"="$udevice"


[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Ethman]
   "Prefix"="ETM"
   "Dll"="ethman.dll"
   "Index"=dword:1
   ; WZCSVC must be started before ethman
   "Order"=dword:2A
   ; Flags==12 is DEVFLAGS_LOADLIBRARY and DEVFLAGS_LOAD_AS_USERPROC
   "Flags"=dword:12
   "UserProcGroup"=dword:3 ; // default to group 3
```

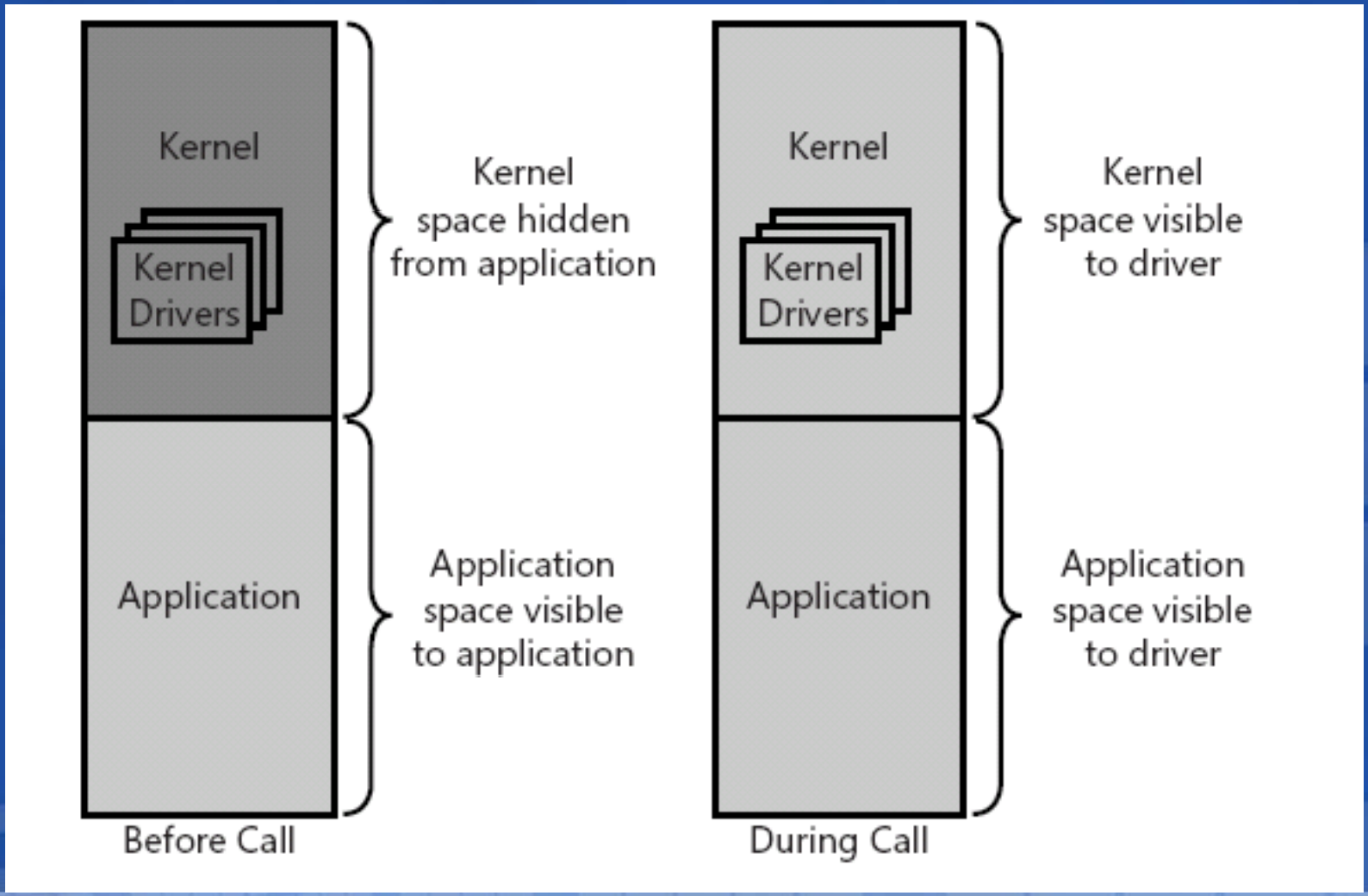
BIB文件的改变

- MODULES 节中：
  - K表示fixup到内核态
  - 没有K则表示用户态
  - Q表示内核态用户太均可
- FILES节中：
  - 可以被加载到用户模式和内核模式

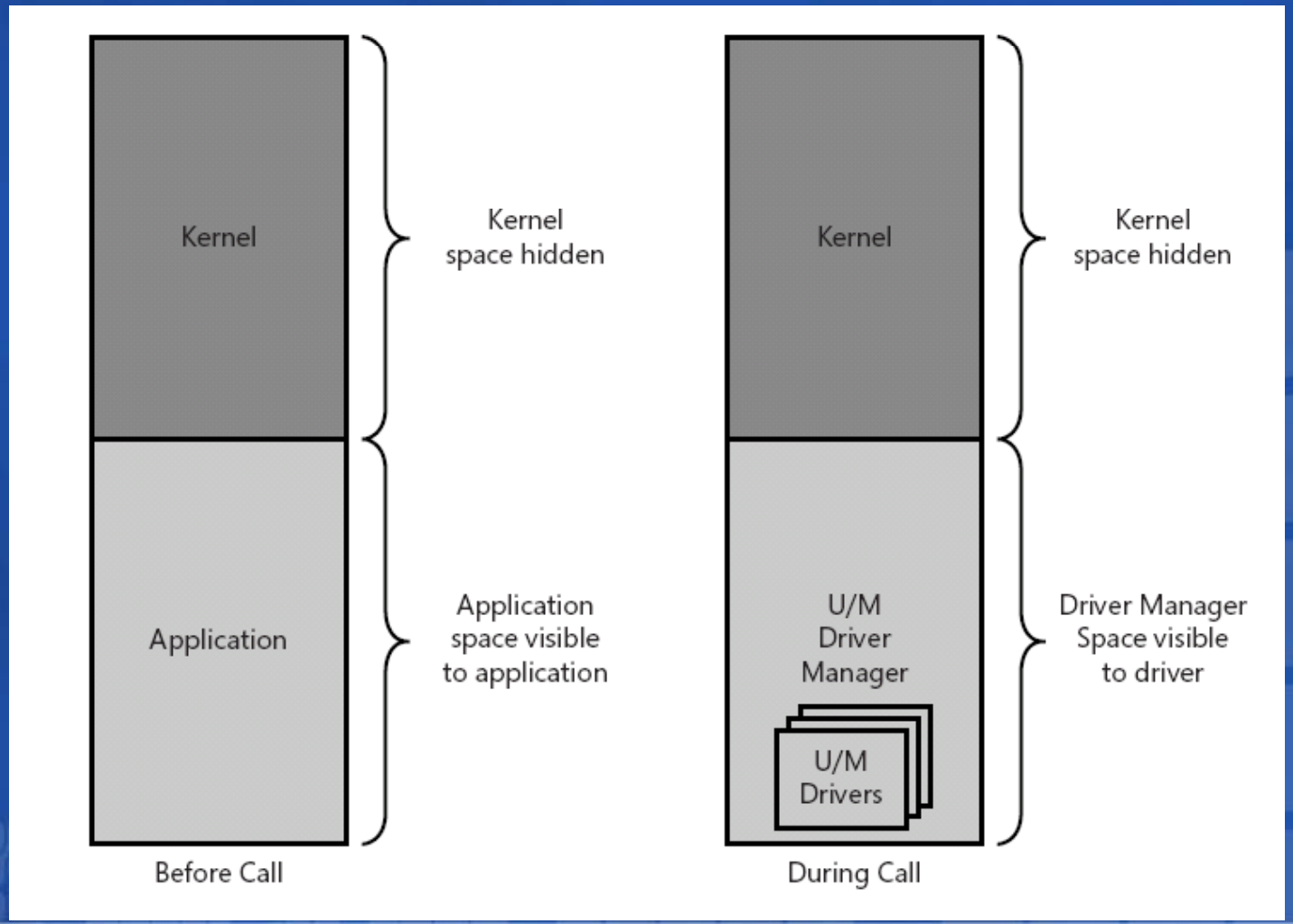
深入理解：
请参考共享源代码

%_WINCEROOT%\PRIVATE\WINCEOS\COREOS\DEVICE

缓冲区管理原因——内核驱动



缓冲区管理原因——用户驱动



缓冲区管理的内容：

- 权限检查
  - 确定是否有权限访问缓冲区
- 列集（Marshalling）
  - 驱动用列集后的指针来访问调用者缓冲区
- 安全拷贝（Secure Copy）
  - 复制缓冲区，避免异步访问出错
- 指针参数
  - 指针作为API的参数
- 内嵌指针
  - 嵌套在结构体中的指针参数

指针参数与嵌套指针

```
// 应用程序代码

typedef struct _mystruct
{
        BYTE *          pBuf;   // buffer
        ULONG           nSize;  // size of the buffer
}InputPara;

WriteFile（hDrv , pBuffer, sizeof(InputPara));
```

```
// 驱动程序代码

DWORD XXX_Write (DWORD hOpenContext, LPCVOID pBuffer,
DWORD Count);
```

处理调用者缓冲区– 列集方法

- 直接访问
  - 调用者缓冲直接可以访问
  - 只有内核态驱动同步访问时才适用
- 拷贝
  - 把调用者的缓冲复制一份
  - 对复制后的缓冲进行操作
  - 最后把这份缓冲复制回去
- 别名
  - 在驱动程序中把调用者缓冲再映射一份
  - 访问该份缓冲会直接影响调用者的原始缓冲

列集之——同步或异步访问

- 同步访问
  - 内核会自动处理参数中的指针
  - 使用**CeOpenCallerBuffer** 函数访问和列集嵌入到结构体中的指针
- 异步访问
  - 最初操作与同步访问一样
  - 使用**CeAllocAsynchronousBuffer** 把指针以异步的方式列集

处理调用者缓冲区– 安全拷贝

- 把输入缓冲区复制到驱动本地
- 然后使用本地缓冲区
- 可防止调用者无意的修改了输入缓冲区
- 性能会降低

安全拷贝的方法

- 手工拷贝
- **CeOpenCallerBuffer**
  - 对嵌入式指针有效
  - ForceDuplicate标志
- **CeAllocDuplicateBuffer**
  - 对指针参数有效

处理缓冲区的API

- CeOpenCallerBuffer
  - Use on embedded pointers
  - Returns checked, marshalled pointer
  - Can force buffer duplication (Secure Copy)
- CeCloseCallerBuffer
  - Frees resources allocated by **CeOpenCallerBuffer**
  - Write back to caller buffer if necessary

处理缓冲区的API

- CeAllocAsynchronousBuffer
  - Allocate driver buffer for asynchronous use
  - Source buffer must be marshalled
  - Must be called synchronously
- CeFreeAsynchronousBuffer
  - Frees resources allocated by CeAllocAsynchronousBuffer
  - Write back to caller buffer if necessary

处理缓冲区的API

- CeAllocDuplicateBuffer
  - Secure copy input buffer
  - Use with pointers only
  - Use with synchronous access only
- CeFreeDuplicateBuffer
  - Frees resources allocated by CeAllocDuplicateBuffer
  - Write back to caller buffer if necessary

用户态驱动程序的限制

- 尽量不使用嵌入到结构体中的指针
  - Reflector 自动映射参数，但不映射嵌入式指针
  - 如果是内核访问该驱动，则无法引用嵌入指针指向的数据
  - 把所有的数据放入单层平坦缓冲区
- 不要支持异步内存访问
  - 指针参数并不会正确的列集

用户态驱动程序的限制

- 谨慎使用内核API
  - 部分API完全不能使用
  - 部分API被Reflector支持，但是可以通过注册表改变配置
- 某些驱动必须运行在内核态
  - Display
  - Networking

内核态驱动的限制

- 无法直接显示UI
  - 需要UI Proxy驱动程序支持
  - 使用 **CeCallUserProc**

	内核模式驱动		用户模式驱动	
	同步	异步	同步	异步
参数列表中的指针	不需要使用	CeAllocDuplicateBuffer CeFreeDuplicateBuffer	不需要使用	CeAllocDuplicateBuffer CeFreeDuplicateBuffer
嵌入到结构体中的指针	不需要使用	CeOpenCallerBuffer CeAllocAsynchronousBuffer CeFreeAsynchronousBuffer CeCloseCallerBuffer	CeOpenCallerBuffer CeCloseCallerBuffer	CeOpenCallerBuffer CeAllocAsynchronousBuffer CeFreeAsynchronousBuffer CeCloseCallerBuffer

深入理解：
请参考共享源代码

%_WINCEROOT%\PUBLIC\COMMON\OAK\INC\Marshal.hpp

Windows CE Base Team Blog:
Marshal Helper API:
http://blogs.gotdotnet.com/ce_base/archive/2006/11/22/marshalling-helper-apis.aspx

```
typedef struct {
        int nSize;
        PBYTE pData;
} BUFDAT, *PBUFDAT;
typedef struct {
I       int nBuffs;
        BUFDAT bd[MAX_BUFFS];
} CHKSUMSTRUCT, *PCHKSUMSTRUCT;

DWORD xxx_IOControl (DWORD dwOpen, DWORD dwCode, PBYTE pIn,
DWORD dwIn, PBYTE pOut, DWORD dwOut, DWORD *pdwBytesWritten)

PCHKSUMSTRUCT pchs = (PCHKSUMSTRUCT)pIn;

for (i = 0; (i < pchs->nBuffs) && (i < MAX_BUFFS); i++) {
        hr = CeOpenCallerBuffer ((PVOID *)&pData,
        pchs->bd[i].pData,  pchs->bd[i].nSize,
        ARG_I_PTR, FALSE);
        for (j = 0; j < pchs->bd[i].nSize; j++)
                dwSum += *pData++;
        CeCloseCallerBuffer (pData, pchs->bd[i].pData,
        pchs->bd[i].nSize, ARG_I_PTR);
}
```

驱动程序的移植

- 驱动的总体架构未发生改变
  - MDD / PDD
- 移植现有的CE 5.0驱动
  - 每个驱动程序大约需要1～3天
- 主要的改动是驱动如何访问客户内存
  - 不再支持SetKMode和SetProcPermissions
  - 提供CEAppCompat工具检查现有驱动的兼容性

DEMO

使用CEAppCompat工具检查应用程序的兼容性。

位置：WINCE600\PUBLIC\COMMON\OAK\BIN\I386

总结

- CE 6内核架构改变引起的驱动程序变化。
- 用户模式驱动与内核模式驱动的不同
- 如何管理缓冲区
- 如何移植

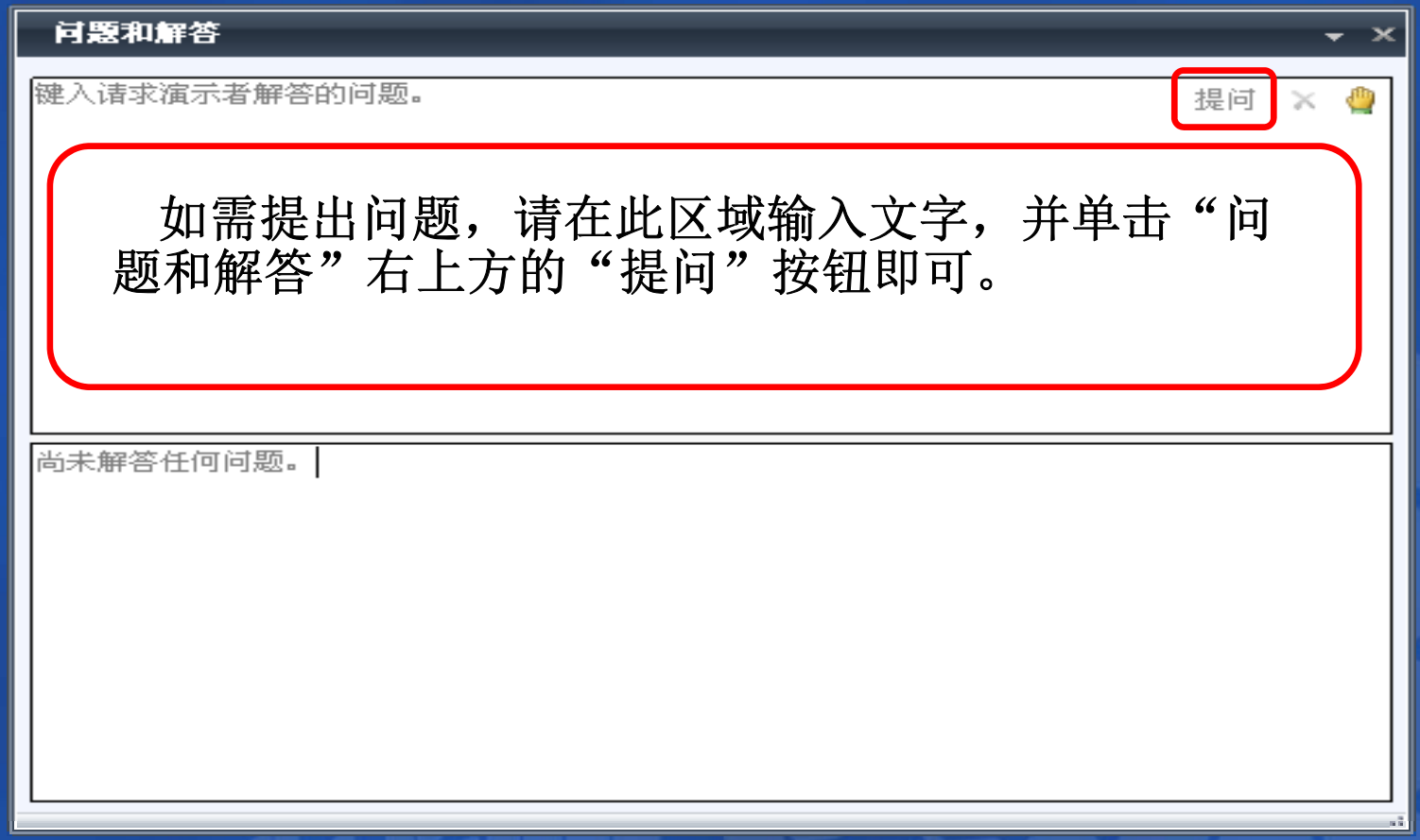
获取更多MSDN资源

您的潜力，我们的动力

Microsoft
微软（中国）有限公司

- **MSDN中文网站**
  http://msdn2.microsoft.com/zh-cn
- **MSDN中文网络广播**
  http:// www.microsoft.com/china/msdn/webcast
- **MSDN免费中文速递邮件 (MSDN Flash)**
  http://msdn2.microsoft.com/zh-cn/flash
- **MSDN开发中心**
  http://msdn2.microsoft.com/zh-cn/developercenters
- MSDN图书中心
  http://www.microsoft.com/china/msdn/book
- 微软嵌入式论坛
  http://forums.microsoft.com/china/default.aspx?ForumGroupID=493&SiteID=15

Question & Answer



您也可以选择在微软中文技术论坛上寻求帮助，**MSDN**中文网络广播的讲师们会定期在论坛上为大家解答与课程相关的技术问题。
http://forums.microsoft.com/china


您的潜力，我们的动力
Microsoft
微软（中国）有限公司
Microsoft
msdn
MSDN Webcasts