10 anni di Windows Embedded

#### di [Beppe Platania](https://mvp.support.microsoft.com/profile=1C0BBF0F-D101-443A-9230-E9D52D2E827A)

* 1. 

*Ripubblicato in Giugno 2013*

* 1. 

Sono ormai passati 10 anni da quando Microsoft ha deciso di entrare nel mondo dei sistemi operativi embedded e vale la pena tracciare un breve excursus storico di questo periodo che ha segnato il passaggio dalla prima versione di Windows CE fino ai nostri giorni, quando i sistemi operativi embedded di Microsoft occupano i primi posti della distribuzione mondiale del settore.

## Dispositivi e sistemi operativi Embedded

Per iniziare, cerchiamo di chiarire cosa intendiamo, in breve, per dispositivi embedded: un apparato elettronico governato da un computer. Chi progetta un dispositivo embedded tende il più possibile ad ottenere un apparato “dedicato” strettamente allo scopo per cui è stato disegnato. Chi utilizza un dispositivo embedded spesso non percepisce neanche la presenza di un computer nel dispositivo.

Esempi classici sono: i mouse, le tastiere e qualsiasi device USB (tutti gestiti, in genere, da microcontrollori a 8-bit), gli hard-disk, i flash-disk, i pen-drive e, fuori dall’ambito dei personal computer, bilance, macchine di misura, strumenti medicali, ecc...

I sistemi Embedded (dedicati) vengono progettati, realizzati e distribuiti per degli obiettivi precisi in contrapposizione ai “sistemi aperti”, tipici dei Personal Computer e non solo, che nascono con la possibilità di essere utilizzati per le più disparate applicazioni.

Altri esempi più immediati sono: le stampanti, i riproduttori DVD, i robot e molti dispositivi simili per l’industria, i navigatori satellitari, i PDA (Personal Device Adapter), i telefoni cellulari, le biglietterie automatiche, i POS (Point of Sale), ecc...

I dispositivi embedded, quindi, hanno la stessa età dei primi processori, anche se per un lungo periodo non è emersa l’esigenza di un vero e proprio sistema operativo. Ogni produttore forniva all’utente il proprio ambiente di sviluppo con svariati linguaggi, oltre naturalmente all’assembler di sistema, per poter sviluppare applicativi e sottosistemi in grado di gestire direttamente il dispositivo. Per una larga fascia di microcontrollori a 8bit e di CPU a 16bit, la situazione non è cambiata di molto, mentre per altri, con l’aumentare della complessità delle richieste, della necessità di adeguarsi agli standard di mercato e la crescente esigenza di mutua comunicazione, si è andata concretizzando una richiesta di sistemi operativi embedded che coronassero esigenze precise:

Affidabilità;

Il dispositivo embedded, dedicato ad un numero finito di applicazioni, ha bisogno di avere un sistema operativo che funzioni sempre, possibilmente con un breve tempo di entrata in funzione all’accensione.

Si deve inoltre tenere presente che molti di questi dispositivi non prevedono aggiornamenti di sistema “sul campo” (palmari, navigatori, controllori di processo industriali, ecc).

Performance;

Ai dispositivi embedded vengono richieste alte prestazioni rispetto alle ridotte risorse di sitema (CPU, flash, RAM, alimentazione, ecc...); ad esempio, ci sono applicazioni embedded che necessitano di dispositivi e sistemi operativi “hard-real-time”. Con questo termine si intende che tutto il sistema, dall’hardware all’applicativo, deve essere in grado di gestire “eventi” in un tempo determinabile a priori, senza che ci possa essere alcuna perdita di informazioni. Per ciò che riguarda il sistema operativo, questo si traduce nella capacità di gestire, in modo completo, la priorità dei processi con una metodologia che sia in grado di controllare tutte le combinazioni di priorità possibili. I processi dovranno essere organizzati e scritti tenendo ben presente il funzionamento del sistema.

Economicità;

I prodotti di pubblico consumo devono avere un prezzo competitivo. Inoltre, l’evoluzione richiesta in molti ambienti embedded esige un rapida reattività alle novità del mercato. Da qui l’esigenza di ambienti di sviluppo semplici, familiari e performanti, sia per il sistema operativo che per gli applicativi.

Un dispositivo embedded ha proprio bisogno di un sistema operativo?

Un dispositivo embedded, per definizione, svolge delle funzioni previste e certificate dal costruttore prima del rilascio. Queste non necessitano sempre di un sistema operativo per poter essere coordinate e per operare correttamente. Questa esigenza si manifesta quando si presentano una o più di queste necessità:

sincronizzazione, schedulazione in condizioni di multitasking

gestione di un gran numero di dispositivi di I/O (di tipo diverso);

gestione dei file, dispositivi di rete, dispositivi video grafici;

gestione della memoria, della sicurezza, dell’alimentazione.

Le risorse del dispositivo, in questi casi, sono più facilmente controllabili utilizzando un sistema operativo.

Prima di percorrere questi 10 anni di Windows Embedded, mettiamo ancora in evidenza alcune particolarità dei prodotti embedded: prendiamo come esempio un chiosco informativo. Esso deve essere realizzato cercando di prevedere tutto ciò che può servire al dispositivo già in laboratorio:

l’aggiornamento guidato: il prodotto dovrebbe essere ri-certificato in laboratorio per qualsiasi modifica o aggiornamento, sia questo del sistema operativo, degli applicativi sviluppati internamente o di quelli di terzi integrati nel prodotto;

l’affidabilità: il progetto del prodotto deve curare l’affidabilità in tutte le fasi di lavoro, di aggiornamento e anche di situazioni prevedibili (cadute improvvise di tensioni, aggiornamenti parziali, ecc...), in modo che il sistema sia mantenuto operativo il più a lungo possibile;

la reperibilità: il materiale (sia hardware che software) utilizzato per il prodotto deve rimanere reperibile per tutta la durata della vita del prodotto. Microsoft garantisce 10 anni di distribuzione per ognuna delle release dei sistemi operativi Windows Embedded;

Un’altra osservazione importante sui che ci può far capire molte scelte architetturali è che il destinatario oggetti dei prodotti embedded NON è l’utente finale, ma è sempre un “produttore” che, una volta adattato e personalizzato il sistema operativo per le proprie esigenze, crea un “prodotto finale” pronto per il mercato.

Partiamo dalla situazione della metà degli anni novanta:

I processori Pentium avevano appena pochi anni di vita, Windows 95 era appena nato e la maggior parte dei Personal Computer utilizzava MS-DOS come sistema operativo.

In ambito Microsoft non esisteva ancora un’offerta Embedded specifica e lo sviluppo di queste soluzioni veniva effettuato con svariati linguaggi, senza piattaforme generalizzate, basandosi su sistemi operativi “aperti”: Ms-Dos, Windows 3.11, Windows NT.

L’esigenza che si stava diffondendo tra gli sviluppatori Windows era quelle di poter scrivere e certificare anche i propri applicativi per il mondo embedded utilizzando strumenti familiari.

## 1996 – Windows CE 1.00

### Nasce il primo Sistema operativo Embedded di Microsoft: Windows CE 1.00.

Scritto per processori PowerPC, MIPS e Intel, veniva distribuito su 22 CD e venne utilizzato da pochi OEM per realizzare piccoli PC palmari.

La soluzione forniva agli sviluppatori Windows un’ambiente di sviluppo simile a quello standard, che presentava una scelta di chiamate al sistema operativo (API) ricavate da quelle WIN-32.

Windows CE era un sistema operativo “nuovo”, che non era completamente compatibile con Ms-DOS nè con Windows-32. E’ un sistema “leggero”, multithread, con un’interfaccia utente grafica opzionale ed era multipiattaforma cioè, poteva funzionare su dei processori di natura diversa.

Il primo prodotto disegnato per Windows CE fu un palmare con funzioni di “organizer”, con un video e una tastiera ridotte.

## 1997-99 Windows CE 2.xx.

Vengono rilasciate le versioni 2.0 2.01 2.11 e 2.12, utilizzate da un maggior numero di Original Equipment Manufacturer (OEM). Nel sistema operativo furono introdotte molte funzionalità per la connessione di rete che permisero, tra l’altro, la nascita dei primi Thin-client: dispositivi che, sfruttando l’architettura “terminal server”, vengono utilizzati come terminali remoti delle applicazioni Windows.

Con la distribuzione di Windows CE 2.0, gli sviluppatori potevano acquistare un Kit di personalizzazione del sistema (ETK=Windows CE Embedded Toolkit ) che prevedeva soltanto una piattaforma hardware. Successivamente, con i “service pack” 2.11 e 2.12 vennero aggiunte interessanti novità: l’inizio dell’interfaccia grafica, una versione della “command-shell” tipo MS-DOS, una prima implementazione della sicurezza, una versione ridotta di Internet Explorer 4.0, l’IrDa e altro ancora.

## Windows NTe

Nel frattempo si andava diffondendo l’esigenza di un sistema operativo embedded che permettesse l’utilizzo degli applicativi scritti per l’ambiente windows standard senza richiederne la conversione per l’ambiente Windows CE.

Molte aziende avevano risolto il problema “ritagliando” e “configurando” opportunamente i sistemi operativi desktop, in particolare Windows 98, 2000 ed NT, ed altre erano arrivate a creare veri e propri ambienti di “personalizzazione”.

Microsoft coglie questa esigenza e realizza un ambiente di sviluppo per “costruire” un sistema operativo partendo dei “componenti di base” e aggiungendo altri componenti dipendenti dai primi, in modo da poter personalizzare il sistema operativo risultante. L’utente di questo ambiente, gestito da un database, può generare facilmente i propri componenti aggiuntivi, determinarne le dipendenze e combinarli per le più svariate esigenze. Nasce Windows NT Embedded!

## 2000 Windows CE 3.00

Questa versione viene radicalmente ridisegnata con un’architettura “hard-real-time”. Il Platform Builder (l’ambiente per personalizzare il sistema operativo) assume definitivamente una veste grafica, interattiva e comprensiva di tool di sviluppo e di debug.

Il nuovo kernel, che rimarrà fino alla versione 5.0, permette di gestire le funzionalità real-time mediante:

256 livelli di priorità (nelle versioni precedenti erano soltanto 8);

La programmabilità del “quantum” di ogni trhead;

La gestione degli interrupt nidificati;

Naturalmente le novità non si fermarono al kernel, ma continuarono: l’object store fu allargato a 256Mb, la dimensione massima dei singoli file fu portata a 32Mb, vennero introdotti e/o aggiornati i supporti DCOM, PPTP, ICS, RDP 5, ecc ....

Con questo rilascio, si può affermare che Microsoft sia entrata a pieno merito nel mondo dei sistemi operativi “embedded real time”!

Parallelamente, Microsoft rilascia una versione dedicata ai palmari (Pocket-PC) che continuerà il suo sviluppo nel 2001 per integrarsi con l’ambiente dei telefoni cellulari, per dare così inizio alla produzione di software per le esigenze di chi, pur muovendosi sul territorio, cominciava ad essere sempre “connesso”.

## Windows CE .NET 4.xx

Il nome dato a questa release di Windows CE mette in evidenza la nascita di una versione della piattaforma .NET dedicata al mondo “mobile” ed Embedded, chiamata .NET Compact Framework. La versione 4.2 (per ironia la versione 4.0 anche chiamandosi .NET non supportava ancora .NET compact framework) offre molte delle potenzialità del run-time .NET utilizzando una libreria “leggera”, in modo da poter essere integrata anche in prodotti di ridotte capacità di calcolo, disco e memoria, prevalentemente alimentati a batteria.

Il messaggio dei contenuti di questa serie di release si basava sostanzialmente su tre concetti:

La produttività;

Venne prestata attenzione a questa importante caratteristica ad iniziare dal “platform builder” (l’ambiente grafico per lo sviluppo dei driver, la generazione del sistema operativo e, soprattutto, l’ambiente per il debug e il test) che venne esteso con nuovi template di device pre-configurati. Questo ambiente integrato permetteva un sempre più rapido riciclo tra build, test e modifiche ed inoltre, essendo un ambiente multipiattaforma, offriva un cross-compilatore interno per le diverse CPU supportate (ARM, MIPS, SH e X86). Vennero aggiunte due nuove piattaforme di sviluppo per gli applicativi: Visual Studio .NET e Embedded Visual C++ 4.00. Venne migliorato l’ambente di emulazione dei device e nacque nuove iniziative per migliorare la produttività:

l’accesso al codice in forma sorgente (2 milioni di linee di codice disponibili);

rafforzamento della comunità Windows Embedded con gruppi di discussione;

oltre 60 “how-to” (descrizioni esplicative per rispondere alle domande di sviluppi più frequenti);

chat e molti eventi per divulgare i progressi ottenuti.

Consolidamento strutturale;

Tutte le funzionalità del prodotto vennero consolidate ed ampliate: dalla parte “real-time” alle dimensioni del sistema, aumentando la componentizzazione fino a 350 componenti, da una nuova gestione dell’alimentazione del device alla piattaforma per le comunicazioni: TCP/IP, IPV4, IPV6, NDIS 5.1, Winsock 2.0, ecc; dai sistemi di controllo remoto (SMNP) ai supporti standard: ECMA, Bluetooth, UPnP, USB, XML, SOAP; dalla gestione dei file (TFAT, BinFS) ai server FTP, HTTP, RAS, PPTP; per finire con i supporti alla sicurezza: Cerberos, PPTP, PEAP a EAP.

Soluzioni innovative;

Queste release furono anche il risultato del lavoro svolto verso una mutua connessione dei device, soprattutto per le soluzioni wireless. La frase emblematica di quel periodo era “connessione per tutti i device in tutti i luoghi in ogni momento (any where, any device, any time)”. Più in particolare, ci riferiamo alla scalabilità delle tecnologie wireless (PAN, LAN, WAN, Bluetooth, 802.11, ecc), all’inserimento nella distribuzione di una ricca piattaforma multimediale (WM 9, DirectX8, I.E. 6.0) e dei viewer dei documenti fondamentali (Excel, Word, PPT, Image, PDF), al supporto multi-lingue per 12 linguaggi, al client RDP 5.1, all’estensione dei device driver (UPnP, 1394, ATA/IDE) e della piattaforma Real time per le comunicazioni IP (SIP, RTC).

## Windows CE 5.xx

Il messaggio principale di questa release è la ricerca della qualità di ogni componente. In quest’ottica nascono “Production-Quality OAL (OEM Adaptation Layer)”, Production-Quality BPS (Board Support Packages)” e “Production-Quality driver”, nel senso che la distribuzione effettuata da Microsoft offre agli utenti pacchetti certificati e, per aiutare gli utenti stessi a certificare i propri sviluppi, vengono creati e distribuiti dei tool di test (CETK = Windows CE Test Kit). Gli aggiornamenti tecnici sono molteplici sia dal punto di vista della piattaforma di sviluppo che da quello del sistema operativo e dei nuovi componenti.Ne citiamo soltanto degli esempi:

Ambiente di sviluppo per driver e build (Platform builder);

Possibilità di mettere dei breakpoint mentre il dispositivo sta lavorando, semplificazione della connessione di debug utilizzando il KITL (Kernel Indipendent Transport Layer), unificazione dell’interfaccia grafica con la gestione a riga comando, nuove opzioni di compilazione e di link, ecc.

Elementi di sistema;

Interrupt di sistema portati da 32 a 64, creazione del thread di power-down, possibilità di variare la frequenza di schedulazione, Embedded Database (EDB), integrazione di tecnologie grafiche e multimediali (Windows Media, Direct3D Mobile, Direct sound,ecc), ecc.

Nuovi componenti;

internet explorer 6.0 per Windows CE, una serie di componenti per la sicurezza (crittografia, gestore delle credenziali, servizi di autenticazione, ecc), aggiornamento delle shell di sistema, Voice Over IP, RTC (real-time communication client), ecc.

## Windows Embedded CE (6.0)

Le due maggiori novità di questa release sono il nuovo kernel che ha permesso di superare alcune barriere sul numero dei processi e la disponibilità di memoria per processo e l’integrazione del Platform Builder nell’ambiente Visual Studio .NET 2005.

Vediamo quali sono state le più importanti novità:

a livello di sistema;

2Gb di memoria virtuale per ogni processo, 32000 processi (fino alla release 5.0 erano 32!!); alcuni componenti critici di sistema sono stati spostati all’interno del kernel; i driver possono essere scritti per lavorare in “kernel mode” con un incremento delle performance o in “user mode” per accrescere la robustezza del sistema, ecc.

A livello di tool per lo sviluppo e il test;

L’integrazione in Visual studio .NET 2005 ha portato immediati benefici: aggiornamento dei compilatori all’ultima release, miglioramento delle performance, aggiunta di editor dedicati (BIB e REG), runtime image viewer, ecc.

Ulteriore disponibilità dei codici sorgente:

Il codice sorgente presente tramite la licenza shared source arriva al 100% del kernel e del device manager, che fino alla release precedente NON erano disponibili.

Nuovi componenti per le nuove tecnologie.

Proiettore wired e wireless per i notebook con Windows Vista (networked projector e remote display); funzionalità legate alla telefonia cellulare (Cellular Network Support): RIL e TAPI, GSM, GPRS, 3G, SMS, ecc; funzionalità multimediali (Network Media Device): il “middleware” NMD, la WM DRM 10 (Cardea) per la compatibilità con PlayForSure, componenti per il motore DVR in MPEG2, ecc.

## Windows XP Embedded

Passiamo al 2001 e l’interesse di Microsoft per i sistemi operativi Embedded diventa così marcato che passano soltanto 6 mesi tra il rilascio di Windows XP a quello del suo analogo per il mondo embedded. Il nome dell’intero pacchetto Windows XP embedded: “Target Designer” (TD) viene preso dal suo elemento più significativo, ma la suite completa è composta da tre ambienti grafici, oltre al TD troviamo il “Component designer” e il “Component Database Manager”. Il prodotto è l’evoluzione dell’ambiente NT embedded: i “componenti” del sistema operativo XP sono stati riordinati in gruppi, ne sono state stabilite le dipendenze e le loro mutue esclusività e da lì è stato ricreato un database ben articolato in modo da permettere all’utente di creare, molto semplicemente, un sistema operativo Windows XP “ritagliato” sulle proprie esigenze. A tutto questo sono stati aggiunti nuovi componenti specifici per l’ambiente embedded: la personalizzazione grafica, la velocità di accensione, l’aggiornamento, la protezione del sistema, ecc...

Il prodotto, che d’allora è stato aggiornato con tre rilasci sia per mantenere l’allineamento con l’ambiente Windows XP Professional sia per le notevoli migliorie dell’ambiente di lavoro e della granularità dei componenti, ha preso in considerazione tre momenti fondamentali per la costruzione di un ambiente di produzione di un sistema operativo personalizzato:

la creazione della piattaforma hardware;

L’esigenza, in questa fase, è quella di creare una lista di tutti i singoli componenti hardware di cui sarà formato il dispositivo embedded finale e, per ognuno di questi, di reperirne il driver opportuno. I costruttori di hardware che hanno una linea di distribuzione dedicata all’embedded offrono già, insieme ai driver XP, oltre alla lista dei componenti delle proprie piattaforme, il “pacchetto” dei driver da inserire nel database di lavoro. Per le altre innumerevoli combinazioni di componenti hardware che possono essere scelte per la propria piattaforma, Microsoft ha preparato una serie di tool per aiutare l’utente a creare la propria lista. Mediante questa lista ci si procurano i driver (è bene ricordare che i driver utilizzati da Windows XP embedded sono i driver rilasciati dai costruttori dei driver per XP, senza alcuna differenza!!).

La scelta del template software;

Il secondo passaggio riguarda la scelta del tipo di dispositivo che si vuole produrre. L’ambiente presenta una serie di template organicamente precostituiti, che permettono di partire da una situazione consolidata; l’utente può poi inserire o eliminare, secondo le proprie esigenze, altri componenti specifici. In questa fase, ci sarà eventualemente anche l’integrazione dell’applicativo specifico dell’utente all’interno del sistema (es: il produttore di un “chiosco informativo” creerà il componente dell’applicativo “chiosco”).

Le funzionalità embedded;

L’ultimo passaggio riguarda l’assemblaggio vero e proprio del sistema:

Presentazione personalizzata e controllata;

Microsoft autorizza i produttori di dispositivi embedded a sostituire la parte di “presentazione iniziale” del sistema (splash-screen) con un logo diverso che può essere, ad esempio, quello del produttore del software o, in alcuni casi, quello del cliente finale. Inoltre, l’ambiente offre una serie di opportunità per gestire al meglio il fatto che spesso i dispositivi embedded non possiedono tastiera e mouse e, soprattutto l’utilizzatore abituale non può essere il referente delle segnalazioni del sistema: su un chiosco informativo è inutile far apparire un pop-up che richiede un intervento sistemistico quando davanti allo schermo c’è il cliente della banca che aveva richiesto un estratto conto.

Configurazioni di lavoro;

Nell’ambiente embedded, le configurazioni degli utenti del sistema, dei privilegi e degli accessi vanno decisi, in gran parte, a livello di produzione del sistema. In genere, l’esigenza richiede due tipi di utenza finale: quella di chi utilizza l’applicativo per cui viene venduto il dispositivo e quella dell’amministratore del sistema o meglio del manutentore. E’ possibile organizzare il sistema in modo che l’utente del dispositivo non possa in alcun modo alterare il sistema operativo mentre il manutentore, per definizione, deve essere in grado di farlo.

Automatismi di aggiornamento sul campo.

L’ultimo punto riguarda l’aggiornamento sul campo del dispositivo. Per questo, Microsoft ha previsto sia *soluzioni standard* (aggiornamenti automatici e client di ambienti di distribuzione software tipo SMS), per permettere ai sistemi XP embedded di coesistere con sistemi XP professional, sia *soluzioni dedicate*.

## WEPOS (Windows Embedded for Point of Services)

Così come dall’ambiente Windows CE sono state create da Microsoft diverse distribuzioni per le specifiche esigenze di mercato (Windows Mobile per PDA, telefonini, navigatori, Windows CE per l’Automotive, ecc ...), così dall’ambiente Windows XP Embedded Microsoft ha preparato un prodotto specifico per i punti di vendita di servizi chiamato Windows Embedded for Point of Services.

Questo prodotto non ha un ambiente di sviluppo come per Windows XP, ma è stato preparato da Microsoft con tutti i componenti specifici dell’ambiente in cui si opererà: chioschi, POS (Point of Sale), Pompe di benzina, Self-Chechout, ecc.

L’installazione è simile a quella di Windows XP Professional, ma con i benefici di avere un sistema pensato per l’embedded con tutti i driver tipici dell’ambiente: PINPAD, Badge reader, stampanti, ecc e con la piattaforma .NET POS per poterli programmare in modo standard.

## Windows Embedded Server

Sotto questo nome, viene presentata un’intera famiglia di prodotti server. Il concetto è quello dell’embedded, ovvero singoli server dedicati a servizi specifici: gestione file, telecomunicazioni, protezione, ecc. Questi server sono stati derivati da Windows Server 2003 R2, con restrizioni esclusivamente sulla licenza e non sulle funzionalità. Il vantaggio è squisitamente economico: i server embedded costano meno dell’omologo standard edition.

## Sistemi Operativi Desktop Classici per sistemi Embedded

Per “Classici” si intendono i sistemi Windows XP, 2000, NT, 3.1 e 3.11 e il “buon vecchio” MS-DOS 6.22. Per ognuno di questi sistemi, esiste a listino la versione “for embedded Systems”, che è del tutto identica alla versione corrispondente del prodotto in ambiente desktop, con il vantaggio della reperibilità: quasi tutti, infatti, nella versione desktop non sono più in distribuzione. A questo si aggiunge nuovamente il vantaggio economico.

## .NET Micro Framework

Eccoci all’ultimo nato della famiglia Windows Embedded: “.NET Micro Framework”. Ideato dal dipartimento di ricerca di Microsoft per dispositivi con un basso consumo di energia e CPU a 32, 16 e 8 bit, senza una richiesta specifica di MMU (Memory Management Unit). Dedicato ad un’ampia fascia di sviluppatori, venne distribuito da MSN Direct già nel 2004 per orologi SPOT (Smart Personal Objects Technology) e per MSTV set-top boxes. Attualmente viene usato come side-show in alcuni portatili distribuiti con Windows Vista. Windows SideShow è una nuova tecnologia che permette all’utente di visualizzare su un piccolo video sulla parte esterna del portatile delle informazioni anche se il PC è spento o ibernato. .NET Micro Framework SDK è un kit di sviluppo che si “installa” nell’ambiente Visual Studio .NET 2005 per permettere lo sviluppo di applicativi scritti in C# e si basa su un sottoinsieme delle librerie .NET e WPF (Windows Presentation Foundation).

Per approfondimenti visita: <http://www.microsoft.com/MSPress/books/10457.aspx>

## Le versioni disponibili

[**Windows Embedded CE (in inglese)**](http://www.microsoft.com/windows/embedded/eval/wince/getgoing.mspx)

Attualmente Windows Embedded CE è in distribuzione nella versione 6.0 e il 30 aprile scorso è stata rilasciato il Service Pack 1.

[**Windows XP Embedded (in inglese)**](http://www.microsoft.com/downloads/details.aspx?FamilyID=9bdf1dea-a37e-4d25-83df-aabbaa78914f&displaylang=en)

Windows XP Embedded viene distribuito con l’aggiornamento che si chiama FP2007 (Feature pack 2007); l’insieme degli ambienti che lo compongono viene chiamato “Microsoft Windows Embedded Studio”. Per il futuro si prevede un FP2008 che verrà rilasciato presumibilmente dopo l’estate 2007 e conterrà alcuni componenti presenti nella distribuzione di Windows Vista (Internet Explorer 7.0, il nuovo media player, ecc...).

[**Windows Embedded for Point Of Services (WEPOS) (in inglese)**](http://www.microsoft.com/presspass/press/2007/apr07/04-19POSUpdatePR.mspx)

Windows Embedded for Point Of Services (WEPOS) viene distribuito nella versione 1.1 e si prevede un FP2007 prima della fine dell’anno.