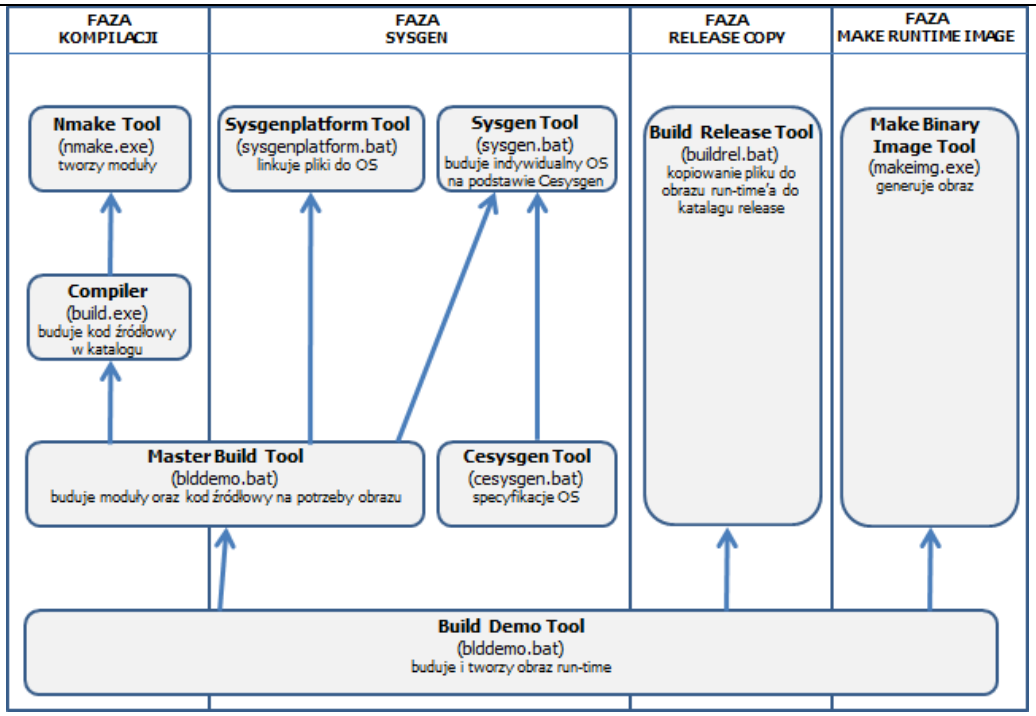
System czasu rzeczywistego	System, w którym wynik przetwarzania nie zależy tylko i wyłącznie od jego logicznej poprawności, ale również od czasu, w jakim został on osiągnięty – jeśli nie są spełnione ograniczenia czasowe, to nastąpił błąd systemu, a wynik przetwarzania nie jest uznawany za prawidłowy Klasyfikacja: • Hard RTOS (ostre ograniczenia czasowe, zadania muszą zakończyć się prawidłowo i w określonym czasie • Soft RTOS (łagodne ograniczenia czasowe, zadania wykonywane najszybciej jak się da, ale nie muszą się zakończyć w ściśle określonym czasie
Osadzenie Windows Embedded 6.0 R2 w urządzeniu wymaga:	• wytworzenia obrazu (run-time image), ktory zawiera: komponenty SO, właściwości, sterowniki, ustawienia konfiguracyjne • wykorzystania Microsoft Platform Builder zintegrowany z VS 2005
Drzewo katalogu ROOT:	• SDK: zawiera kompilatory dla platform x86, ARM, SH4 • OSDESIGNS: domyslnie katalog zawiera OS Designs, które są w trakcie realizacji. Kazy podfolder odwołuje się do designu (projektu) SO. Os Design zawiera moduły: narzędzia i sterowniki • PLATFORM: katalog zawiera część systemu operacyjnego zależnego od sprzętu (hardware dependant), takie jak BSP, podfoldery zawierają OAL • PUBLIC: katalog zawiera komponenty SO niezależne od sprzętu • PRIVATE: zawiera kod źródłowy dla SO • OTHERS: zawiera różne komponenty • 3RDPARTY: komponenty skopiowane z folderów PUBLIC i PRIVATE.
Proces projektowania SO:	• korzystanie z catalog items (wykorzystywanie komponentów i sterowników • dodawanie komponentów w postaci podprojektów • dostosowywanie rejestrów • edycja i modyfikacja ustawień procesu wytwarzania obrazu • wykorzystanie i modyfikacja właściwości BSP
Właściwości projektu, ustawienia regionalne:	• Locales • Default Locales • Code Pages • Localize The Build • Strick Localization Checking In The Build
POSIX	Przenośny interfejs dla systemu operacyjnego Unix. Prace rozpoczęło stowarzyszenie IEEE i dlatego zwany jest również IEEE 1003. POSIX standaryzuje: • interfejs programistyczny (API); • interfejs użytkownika, np. polecenia systemowe takie jak awk, echo, ed; • właściwości powłoki systemowej. Implementacje są w MacOSX10.5, QNX. Rozszerzenia: semafory, proces blokowania pamięci, pamięć współdzielona, kolejkowanie, liczniki czasu, synchroniczne i asynchroniczne operacje we/wy, mutexy.
Jądro systemu	Jest to część systemu wielozadaniowego odpowiedzialna na zarządzanie i komunikację między zadaniami. Podstawowa funkcja jądra to przełączanie zadań (kontekstu). Cechy jądra: • wielozadaniowość, • wielowątkowość, • wielobieżność, • skalowalność, • wywłaszczalność.
Fiber a wątek	Fiber jest lżejszą odmianą wątku. Podobnie do wątków, fiber współdzieli przestrzeń adresową. Jednakże fibery używają współbieżnej wielozadaniowości a wątki wzajemnie się wykluczającej wielozadaniowości. Wątki często korzystają z procesu planisty jądra systemu do zatrzymywania i wszczynania wątków. Fibery podczas wykonywania wykonują kolejnego fibera. Ze względu na te cechy Fiberów, bezpieczeństwo jest mniejszym problemem niż w przypadku wątków i można pomijać synchronizację, ponieważ są zsynchronizowane bezwarunkowo. Problemem jest to, że fibery nie mogą korzystać z dodatkowych procesorów znajdujących się w komputerze bez użycia wątków.
RTLinux	OS hard RTOS posiadający 2 jądra: RTCore + Linux. Architektura wymusza oddzielenie części obliczeniowej od operacji dyskowych i sieciowych, RTCore szybko obsługuje przerwania przy znikomych operacjach dyskowych.
QNX Neutrino	OS hard RTOS posiadający architekturę mikrojądra, modułowa struktura oparta na przesyłaniu komunikatów klient-serwer, możliwość zdeterminowania czasu reakcji na zdarzenia występujące w systemie, małoawaryjny
Windows CE 5.0	OS hard RTOS posiadający architekturę modułową, zoptymalizowany dla urządzeń o niewielkiej

	ilości pamięci (potrzeba 1MB Ram) . Umożliwia uruchomienie 32 procesów, 32MB przestrzeń adresowa procesu,
Windows CE 6.0	OS hard RTOS, 32bit, złożony z komponentów, 4GB pamięci wirtualnej, po 2GB na procesy jądra i użytkowników, SBD, BSP[komponenty oprogramowania służące do dopasowania systemu operacyjnego do współpracy z określonym sprzętem] (OAL [zawiera kod, który tworzy obraz abstrakcyjny jądra systemu operacyjnego niezależnie od sprzętu (fizycznej platformy). Tzn. że jądro WIN Embedded CE może funkcjonować na różnych platformach (kopiować z jednej platformy i odpowiednio zmodyfikować do współpracy z drugą)], sterowniki i pliki konfiguracyjne) API, przestrzeń adresowa 1GB, 32.768 uruchomionych procesów, obsługa 512MB pamięci fizycznej, Device.exe, filesys.exe i GWES.exe przeniesiono do trybu jądra, zarządzanie pamięcią, wielozadaniowy z mech. wywłaszczania zdań
Obiekty do synchronizacji	sekcje krytyczne, mutex, semafory, zdarzenia, point-to-point message queue
Sekcja krytyczna	Zmienna, która pozwala blokować wątki przed jednoczesnym wejście do pewnego obszaru kodu
Mutex	rodzaj semaforów binarnych, realizujący zasadę posiadania: tylko zadanie, które blokuje zasób może ten zasób zwolnić
Semafor	umożliwiają wzajemne wykluczanie i synchronizację zadań, obiekt jądra, który może zostać zajęty przez jeden lub kilka wątków w celu sterowania dostępem do wspólnego zasobu, semafory są globalne: dowolny proces może mieć do nich dostęp
Zdarzenia	pozwalają sygnalizować wystąpienie pewnych okoliczności
Różnica Semafor - Mutex	Wątki nie wchodzą w posiadanie semaforów. W przypadku mutexów, jeśli wątek po raz kolejny zażąda dostępu do mutexu, którego jest właścicielem dostęp otrzymuje natychmiast. Jeśli wątek nagle rozpocznie czekanie na ten sam semafor, to semafor zachowuje się tak, jakby wejścia zażądał każdy inny wątek. Ponadto mutex może być uwolniony tylko przez wątek, który jest jego właścicielem, natomiast licznik semafora może być zwiększony przez dowolny wątek, który z tego semafora korzysta
Elementy obrazu	komponenty SO, właściwości, sterowniki, ustawienia konfiguracyjne
Elementy projektowania SO	• budowanie projektu (wzorce SO) • dostosowywanie projektu SO w oparciu o *catalog components* • konfiguracja kompilacji (*build configuration*) • parametry kompilacji (*build properties*) • konfiguracja zaawansowana • drzewo katalogów projektu
Wzorcowe BSP	ARM, MIPS, SHx, x86
Właściwości projektu	• Zmienne środowiskowe • Łączenie projektu SO z wieloma platformami sprzętowymi • Pliki w katalogu projektu
Podprojekty, typy	• Applications Win32 • Console Applications Win32 • DLL • Static library • TUX dll
Budowanie obrazu systemu operacyjnego	• Budowanie obrazu (*Build run-time image*) • Analiza wyników procesu budowania obrazu oraz plików konfiguracyjnych • Osadzanie (*deploying*) obrazu w urządzeniu
Typy obrazu	DEBUG (WINCEDEBUG=DEBUG, WINCESHIP=0) RELEASE (WINCEDEBUG=RETAIL, WINCESHIP=0) SHIP (IMGNOKITL=1, IMGNODEBUGGER=1, WINCEDEBUG=RETAIL, and WINCESHIP=1)
KITL	jest podstawowym protokołem debugowanie dla urządzeń Windowsa CE. KITL zarządza transportem i strumieniem abstrakcji dla aplikacji. Zapewnia: inicjalizację transportu, podłączenie/odłączenie strumienia, wspólne API
Komponent	niezależnie wytworzony, skompilowany (z ukrytymi szczegółami implementacyjnymi) moduł programowy, udostępniający swą funkcjonalność za pomocą jednoznacznie zdefiniowanego interfejsu, zdolny do współdziałania z większą całością (systemem) oraz innymi komponentami.
Klonowanie	Aby nie modyfikować kodu bazowego komponentów dokonujemy najpierw ich klonowanie i dopiero na kopii dokonywać modyfikacji.
Stany debuggowania	**0** Errors, **1** Warnings, **2** Performance, **3** Temporary tests, **4** Enter Exit, **5** Initalize, **6** Calss, **7** Verbeco, **8** Surface Create, **9** Flp, **10** Line, **11** Hardware, **12** Polygon, **13** Cursors, **14 i 15** nieużywane
Katalog użytkownika	W kluczu rejestru HKEY_LOCAL_MACHINE\init\ BootVars jest zdefiniowany katalog ProfileDir ProfileDir=DocumentsAndSettings

<table>
<tr><td>Fazy budowania</td><td>


Kompilacja: kompilator i konsolidator wykorzystują kod źródłowy i pliki zasobów, żeby wygenerować pliki wykonywalne .exe, statyczne biblioteki .lib, dynamiczne biblioteki .dll, i zasoby binarne .res w zależności od zaznaczonych opcji. Proces ten może trwać nawet do kilku godzin, lecz na szczęście rzadko się zdarza, by trzeba było go powtarzać, ponieważ binaria są udostępnione przez Microsoft. Nie powinno się modyfikować kodu źródłowego znajdującego się w folderach Private I Public.

Sysgen: ustawiane są, bądź czyszczone zmienne SYSGEN w zależności od dołączonych katalogów I drzew zależności do projektu OS, filtrowane pliki nagłówków i tworzone są biblioteki dla SDK zdefiniowanego w projekcie OS, tworzone są pliki konfiguracyjne obrazu run-time oraz budowany jest BSP w oparciu o pliki źródłowe z katalogu Platform.

Build : pliki źródłowe BSP I aplikacje są przetwarzane przy użyciu plików wygenerowanych w fazie Sysgen. W tym momencie budowane są sterowniki sprzętowe i warstwa adaptacji OEM. Pomimo, że procesy zachodzące podczas fazy Build są wywoływane automatycznie z fazy Sysgen, to jeśli zmodyfikujemy tylko BSP I podprojekty, to można przebudować BSP I podprojekty bez ponownego uruchamiania narzędzi Sysgen.

Release Copy: kopiowane są wszystkie pliki wymagane do stworzenia obrazu do katalogu release. W skład ten wchodzą pliki lib, dll i exe stworzone podczas fazy kompilacji, sysgen, oraz obraz bin. .bib I pliki rejestru .reg. Faza ta może zostać pominięta gdy pliki nagłówków I biblioteki są aktualne.

Make Run-Time Image: kopiowane są pliki projektowe (Project.bib, Project.dat, Project.db, and Project.reg) do katalogu wydania i są one łączone w obraz. Dyrektywy zawarte w zmiennych środowiskowych wyspecyfikowane w plikach .reg I .bib określają ktore katalogi są dołączane do obrazu system. Obraz najczęściej się nazywa Nk.bin I można go pobrać I umieścić na docelowym urządzeniu.
</td></tr>
<tr><td>Fmerge.exe</td><td>scalanie plików konfiguracyjnych</td></tr>
<tr><td>Jaką fazę wywołać po modyfikacji w projekcie SO ?</td><td>
<table>
<tr><th></th><th>BSP i podprojekty</th><th>ustawienia obrazu</th><th>add/del katalogu komponentu</th></tr>
<tr><td>Sysgen</td><td>-</td><td>-</td><td>+</td></tr>
<tr><td>Post-Sysgen Build</td><td>+</td><td>możliwe</td><td>+</td></tr>
<tr><td>Buildrel</td><td>możliwe</td><td>+</td><td>+</td></tr>
<tr><td>Makeimg</td><td>+</td><td>+</td><td>+</td></tr>
</table>
</td></tr>
<tr><td>Kiosk Mode</td><td>Wiele urządzeń takich jak monitory medyczne, czy bankomaty są zaprojektowane do wykonywania jednego zadania. W takim przypadku niepotrzebna jest standardowa powłoka graficzna dla takiego urządzenia. Można usunąć standardową powłokę graficzną i ograniczyć dostęp do Panelu Sterowania i zabronić użytkownikom uruchamiania dodatkowych aplikacji, co skutkuje skonfigurowaniem urządzenia w trybie KIOSK, dostosowanym tylko do wykonywania konkretnych zadań bez dostępu do powłoki.</td></tr>
</table>