Porównanie XWindow oraz WindowsAPI, MFC i .NET

Podstawową różnicą w podejściu do obu platform jest otwartość XWindow. Poza tym XWindow charakteryzuje przenośność między platformowa. Co do tego ostatniego pewną konkurencję tworzy .NET, który z założenia też jest niezależny od platformy, choć efektywnie funkcjonuje jedynie na Windowsach.

Weźmy jednak pod uwagę różnice w logice obu projektów. Przede wszystkim XWindow i systemy od niego zależne są podzielone na moduły, a sam XWindow jest jedną z aplikacji w systemie operacyjnym. Rozwiązanie to charakteryzuje struktura klient-serwer. Bazę stanowi XWindow server, który odpowiada za komunikację ze sprzętem. Komunikacja przez sieć między nim a klientami odbywa się transparentnie dla warstw wyższych. Klient odpowiada za wyliczenie zawartości, którą każe za pomocą zestawu prostszych komend wyświetlić serwerowi. W przypadku Windows funkcje graficzne są ściśle związane z systemem, stanowią część Windows API (wraz z funkcjami plikowymi, sieciowymi itd.).

Modularność powoduje również, że w prosty sposób można zmieniać menedżerów okien, co w przypadku systemu MS Windows jest skomplikowane, ponieważ powłoka graficzna (explorer.exe) jest silnie zintegrowana z innymi częściami funkcjonalnymi systemu operacyjnego.

Kolejna różnica polega na różnicach logicznych we wzajemnych relacjach między oknami. XWindow utrzymuje jedynie relacje rodzic-dziecko, Z-index (kolejność pod względem widoczności). Windows API dokłada do tego relację własności oraz możliwość dziedziczenia klas okien.

I w XWindow, i MS Windows, komunikacja systemu z aplikacją odbywa się za pomocą kolejek zdarzeń/zleceń/wiadomości (Event - server->aplikacja, Request – aplikacja->server, Message). W przypadku XWindow kolejka związana jest z połączeniem nawiązanym z danym wyświetlaczem (Display). Ponieważ pobranie kolejnego zdarzenia poprzez np. funkcję XNextEvent pobiera w parametrze identyfikator wyświetlacza, nie ma korelacji wątek-kolejka. Inaczej jest w MS Windows – tutaj przypada jedna kolejka na wątek. Konsekwencją jest potrzeba realizacji w aplikacji dla XWindow pętli, która będzie przechwytywać zdarzenia o wszystkich oknach, stworzonych przez wszystkie wątki aplikacji dla danego wyświetlacza. W MS Windows do danego wątku trafiają tylko informacje o stworzonych przez niego oknach.

W obu systemach jest możliwe włożenie przez aplikację zdarzenia/wiadomości do kolejki. Można wykorzystać ten mechanizm do komunikacji kontrolki z oknem rodzicem.

Windows API pozwala na niekontrolowane przesyłanie komunikatów między oknami, co pozwala np. na odgwiazdkowywanie haseł wpisanych w EditBox jednej aplikacji przez inną aplikację. Kwestia ta jest tym poważniejsza, że taki komunikat nie jest weryfikowany nawet, gdy jest przesyłany między aplikacjami uruchomionymi przez dwóch różnych użytkowników. W systemie XWindow przesyłanie zdarzeń pomiędzy klientami jest określone przez standardy ICCC.

W obu systemach brak obiektowości. Jest ona emulowana przez biblioteki na wyższym poziomie. W XWindow np. XtIntrinsics, a w MS Windows np. MFC. Dopiero Windowsowy .NET proponuje całkowite odcięcie od wywołań Windows API w zamian oferując czysty, obiektowy interfejs.

Biblioteka MFC jest zdominowana przez różne konwencje tworzenia aplikacji, takie jak struktura dokument-widok, SDI, MDI, automatyczne powiązanie stanu kontrolek z bieżącym stanem aplikacji. Podczas gdy biblioteki wywodzące się z systemów UNIX udostępniają raczej narzędzia, niż konwencje.