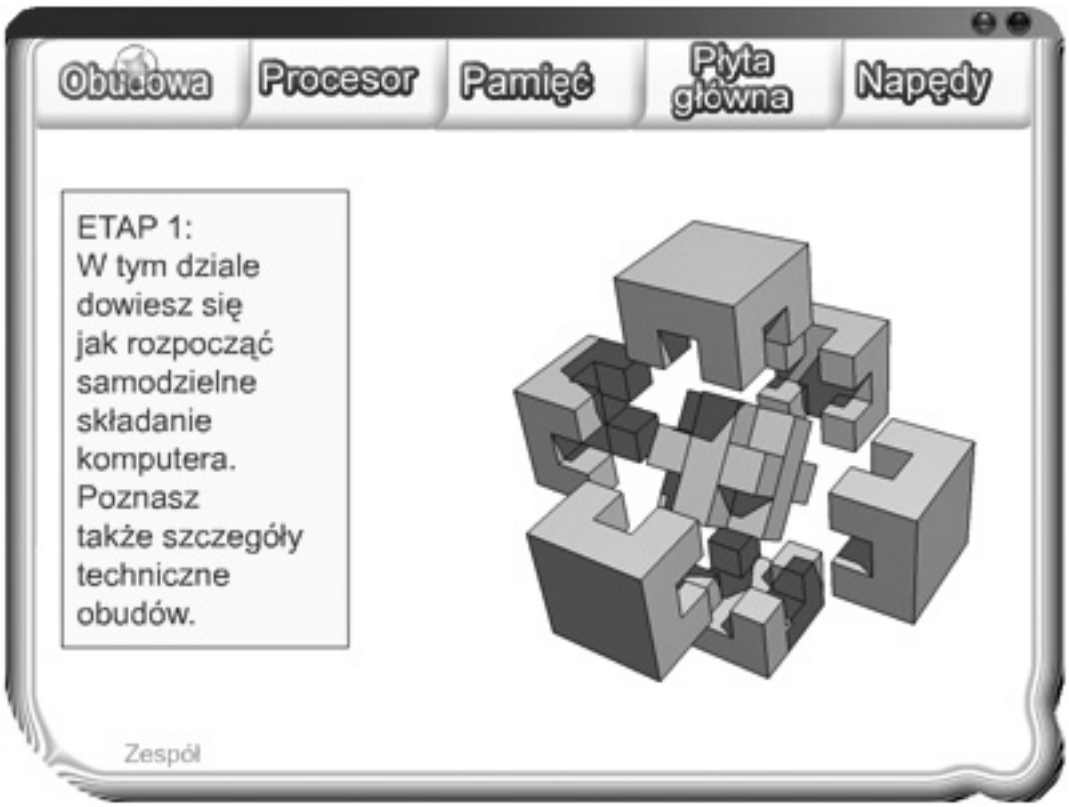
Projekt Inżynierski
"Składacz"
Multimedialny Program Do Nauki
Składania Komputerów

Jarosław Adamski, Marcin Jędrzejewski, Bogdan Sobczak

1 października 2002 roku

Spis treści

Część I
Projekt od strony koncepcyjnej



Rysunek 1: Okno główne aplikacji

1 Wstęp

Ze względu na postępującą informatyzację życia, coraz więcej ludzi zaczyna korzystać z komputerów. Większość ludzi nie interesuje się ich budową. Rozbudowę najczęściej pozostawiają firmom serwisowym, lub po prostu zakupują nowy sprzęt - często przepłacając. Aby nauczyć się jak bezpiecznie rozbudowywać komputer, często trzeba wiele eksperymentować. Może to się okazać niebezpieczne zarówno dla sprzętu jak i dla eksperymentatora.

Takie czynności jak wymiana pamięci, dodanie nowego dysku twardego, czy chociażby odkurzenie płyty głównej, są na tyle proste, że każdy powinien być sam w stanie je robić. Wiele osób ma jednak przed tym opór. Dotyczy to przede wszystkim osób, które zaczęły w dość późnym wieku korzystać z komputerów.

Oczywiście są książki ze szczegółowymi opisami oraz rysunkami pokazującymi krok po kroku jak należy postępować, ale często są one mało przystępne albo po prostu niedostępne. Księgarnie informatyczne najczęściej pełne są literatury technicznej dla zaawansowanych. Książki takie często zniechęcają do eksperymentowania, a także zabierają mnóstwo czasu.

Ciekawym pomysłem okazuje się więc, aplikacja multimedialna. Możliwość obejrzenia całej procedury np. wymiany kości pamięci jest o wiele bardziej przystępna niż przeczytanie na ten temat

rozdziału w książce czy artykułu w gazecie. Podobnych programów na rynku nie ma zbyt dużo.

Nasz projekt wyróżnia się nowatorskim interfejsem użytkownika oraz wysoką jakością tekstów, zdjęć i filmów.

2 Interfejs użytkownika

Obecnie programy są najczęściej tworzone z przeznaczeniem dla graficznych systemów operacyjnych takich jak Windows. Są one wyposażane w graficzne interfejsy użytkownika. W przeszłości aplikacje były często tworzone z przeznaczeniem dla trybu tekstowego. Wymagały one pewnego stopnia zaawansowania od użytkownika. Taka osoba musiała spamiętać dużą ilość komend, a także całkiem nieźle pisać na klawiaturze. Takie programy są nadal popularne, ale głównie w środowiskach unixopodobnych. Nie oznacza to, że Windows całkowicie pozbył się trybu tekstowego. Na przykład taki program jak AutoCad umożliwia wprowadzanie komend tekstowych. Jednocześnie komendy te mają swoje odpowiedniki w opcjach menu oraz różnych okienkach dialogowych.

Program z graficznym interfejsem użytkownika ma wiele zalet. Po pierwsze użytkownik może natychmiast go zacząć używać. Na przykład, na rynku jest dostępnych wiele firm tworzących programy do składu tekstu. Mimo że firmy się różnią to przejście z jednego programu do drugiego nie sprawia żadnych trudności. Praktycznie wszystkie te programy oferują podobne opcje, ale pod trochę zmienionym interfejsie użytkownika. Kolejną zaletą takich programów jest możliwość wykorzystywania myszy do interaktywnego tworzenia dokumentów, rysowania rysunków, wydawania poleceń. Zmniejsza to ilość popełnianych błędów, co często się zdarza przy komendach podawanych w trybie tekstowym. Pola edycyjne mogą np. akceptować tylko liczby z danego przedziału, i natychmiastowo informować użytkownika o błędzie. Również istotną rolę może odgrywać kształt kursora myszy, który dynamicznie się zmienia w zależności od stanu, w jakim znajduje się aplikacja.

2.1 Rola, jaką odgrywa

Głównym zadaniem interfejsu użytkownika jest zapewnienie wymiany informacji między człowiekiem a komputerem. W przypadku aplikacji multimedialnej oznacza to łatwy i przystępny sposób wybierania i prezentowania danych. Programy multimedialne różnią się od innych aplikacji użytkowych. Użytkownicy mają większe wymagania od nich i oczekują często oryginalnego podejścia do prezentacji. Podczas gdy programy typu arkusze kalkulacyjne albo procesory tekstu lepiej działają, gdy używają standardowego interfejsu użytkownika charakterystycznego dla danego systemu operacyjnego, programy multimedialne wykorzystują odmienne sposoby na interakcję z użytkownikiem. Natywne interfejsy użytkownika są najczęściej wykorzystywane głównie po to, aby aplikacja nie odróżniała się od innych. Jeśli używa się wielu programów w codziennej pracy - co często się zdarza - wówczas różnorodność interfejsów może być męcząca. Programy multimedialne natomiast rzadko są wykorzystywane w połączeniu z innymi programami. Często służą one do rozrywki i dlatego jest im bliżej do gier komputerowych niż do programów użytkowych typu arkusze kalkulacyjne albo procesory tekstu.

2.2 Zalety dobrego interfejsu

Twórcy aplikacji multimedialnych starają się tworzyć swoje produkty tak, aby były jak najbardziej przyjazne użytkownikowi. Określają oni, które zmysły będą najodpowiedniejsze do odbierania danej informacji. Na przykład, aby nauczyć studenta rozwiązywania równań całkowych najlepszym podejściem wydaje się być po prostu pokazanie na przykładach poprawnego toku rozumowania.

Natomiast jeśli chodzi o powiedzmy montowanie silnika w samochodzie, to najlepszym sposobem prezentacji byłby tu film. Twórcy programów multimedialnych najczęściej starają się działać na zmysł wzroku, słuchu i dotyku. Przykładowo użytkownik może zobaczyć film oraz usłyszeć ścieżkę dźwiękową, a w trakcie odtwarzania może używając myszy zatrzymać go.

Ponieważ istnieje wiele różnych technik tworzenia multimedialnych interfejsów użytkownika, powstały również pewne ogólne zasady komponowania ich. Mają one pomóc w sytuacjach, gdy interfejs może stać się mało przyjazny dla użytkownika.([2])

- **Prostota** - interfejs użytkownika powinien być jak najbardziej uproszczony.
- **Bycie konsekwentnym** - używanie podobnych metod prezentacji w całym produkcie, ma to na celu ułatwienie użytkownikowi nauki nawigacji po programie.
- **Pozwól użytkownikowi kontrolować interakcją** - użytkownik powinien mieć pełną kontrolę nad stanami w jakich mogą się znajdować poszczególne moduły aplikacji. Przykładowo użytkownik powinien mieć możliwość w każdej chwili zastopować odtwarzany film, przerwać odtwarzanie dźwięku albo po prostu go wyciszyć. Nie powinno się tu niczego zakładać - tylko dać wolną rękę użytkownikowi.
- **Sprzężenie zwrotne** - należy reagować na każde polecenie użytkownika, inaczej może on zacząć podejrzewać że program albo komputer się zawiesił.
- **Nieobciążanie pamięci krótkotrwałej** - okienka dialogowe powinny być tak zaprojektowane, aby nie wymagały od użytkownika zapamiętywania zbyt wielu pól edycyjnych albo poprzednich kroków jeśli np. jakaś procedura podawania danych jest długotrwała.

Psychologiczna reguła 7 ± 2, mówi że liczba obiektów, nad którymi człowiek może się jednocześnie skoncentrować wynosi między 5 a 9. Oznacza to, że projektant interfejsu użytkownika powinien ograniczyć liczbę różnych elementów dialogów właśnie do tej liczby.

Stworzenie dobrego interfejsu użytkownika nie jest, łatwą sprawą. Systemy operacyjne umożliwiają ujednolicenie wyglądu aplikacji, przez co sprawiają że programy stają się jednolite i często wręcz męczące. Interfejs użytkownika aplikacji multimedialnej nie może wzbudzać takich emocji u użytkownika. Powinien być oryginalny, ale powinien także spełniać wyżej wymienione cechy.

2.3 Przekaz multimedialny

Nauczanie poprzez programy multimedialne na CD-ROM'ach ma wiele zalet. Najważniejszymi z nich są :

1. odbywa się własnym tempem
2. jest interaktywne
3. jest stosunkowo tanie

Przy tradycyjnych metodach przekazywania wiedzy, istnieje zawsze współczynnik wprowadzający stres. Może to być na przykład nauczyciel, którego wymagania przerastają możliwości ucznia. Może to być także kiepsko napisana książka. Programy multimedialne umożliwiają przekazywanie tej samej wiedzy, ale pod różnymi postaciami. Najczęściej jest to możliwość oglądania filmów, słuchania opisów albo czytania bogatych w ilustracje tekstów. Teksty są często wzbogacane w połączenia i

odnośniki umożliwiające przejście do innego podobnego tekstu. Często w tego typu programach jest możliwość w tym samym czasie słuchania lektora, czytania tekstu a także oglądania filmu.

Programy multimedialne nie nakładają żadnych ograniczeń na formę wykonania. Student może przybierać rolę np. animowanego bohatera, który przedziera się przez świat matematyki, fizyki czy astronomii. Można w ten sposób pokazać uczniom, że nauka może być zabawna, i w ten sposób zachęcić ich do zgłębiania wiedzy.

Multimedialne CD-ROM'y umożliwiają także zrealizować naukę w sposób praktyczny. Czyli np. można zasymulować czynności, jakie ma wykonywać sprzedawca w sklepie. Program taki mógłby przypominać grę komputerową, sprzedawca musiałby wydawać reszty albo przeliczać waluty, za co dostawał by punkty. Podobne zastosowania są powszechnie wykorzystywane w medycynie albo w wojsku.

Nauka przy pomocy programów multimedialnych ma też swoje wady. Pierwszą i najpoważniejszą jest brak instruktora, czyli brak kontaktu z kimś, kto posiada doświadczenie w danej dziedzinie. Program nie jest w stanie w pełni zasymulować wszystkich elementów związanych z procesem uczenia się. Jednym z takich elementów jest czynnik ludzki. Nauka przychodzi łatwiej, jeśli jest się w większej grupie w której można wymieniać swoje poglądy i doświadczenia. Uczeń będąc w grupie nabiera większej motywacji oraz uczy się jak nawiązywać znajomości, oraz jak pracować w grupie.

Trzeba pamiętać, że programy multimedialne nie mogą stanowić jednego źródła wiedzy. Powinny stanowić dodatek do tych tradycyjnych. Ponieważ działają one na wszystkie zmysły człowieka, sprawiają tym samym że nauka jest bardziej efektywna.

2.4 Przyszłość

Komputery z roku na rok stają się coraz bardziej wydajne. Gordon Moore ponad 20 lat temu przewidział, że co 18 miesięcy będzie podwajać się gęstość upakowania tranzystorów w układach elektronicznych, a wraz z tym będzie rosła moc obliczeniowa procesorów i malała ich cena. Tendencja ta jest jak najbardziej prawdziwa. Wraz z maleniem cen wytwarzania procesorów, okazało się opłacalnym wyposażenie komputera w bardziej zaawansowany procesor na karcie graficznej. Tak zwane procesory graficzne (GPU - graphic processing unit) stają się coraz bardziej popularne. Karty graficzne wyposażone w takie układy stają się coraz częstszym elementem nowych komputerów.

Zaletą tych układów jest możliwość odciążenia procesora od wykonywania czasochłonnych często operacji graficznych. Tak jak kiedyś koprocesor służył do wykonywania obliczeń zmiennopozycyjnych, tak teraz procesory graficzne wykonują renderowanie milionów trójkątów na ekranie. Główne mikroprocesory są często też wyposażane w zestawy instrukcji (np. MMX, 3dnow!, ISSE) przeznaczonych specjalnie dla zastosowań w animacji 3D (np. przekształcenia macierzy).

Wszystko to oznacza, że twórcy interfejsów użytkownika mogą czuć się mniej skrępowani. Wcześniej poważnym ograniczeniem były względy wydajności aplikacji. Nie można było sobie pozwolić, aby komputer większość czasu wykonywał obliczenia potrzebne do wyświetlania rozbudowanego systemu okienek czy kontrolek. Dodatkowo jeśli system operacyjny był wielozadaniowy to wiele aplikacji działających równocześnie mogło by spowodować znaczne spowolnienie pracy systemu.

Zalety wymienionych powyżej kart graficznych wykorzystują obecnie głównie tylko programiści gier komputerowych. Jeśli chodzi o zwykłe programy użytkowe, nadal najczęściej korzysta się ze standardowo wyglądających interfejsów użytkownika. Zaczynają jednak pojawiać się pewne próby przejścia z dwu wymiarowego interfejsu do trójwymiarowego. Na przykład 3DNA[1] jest projektem, którego celem jest wprowadzenie trzeciego wymiaru do codziennego użytku w takich zadaniach jak przeglądanie folderów na dysku twardym albo uruchamianie aplikacji.

[1] `http://www.3dna.net/`

Innym ciekawym pomysłem na tworzenie GUI dla aplikacji jest wykorzystanie dynamicznego HTML'a. Dzięki takiemu podejściu użytkownik może samemu dokonywać zmian w rozkładzie kontrolek w np. dialogach. Potrzebna mu jest tylko znajomość HTML'a, co obecnie jest dość poszechne. Przykład można znaleźć pod tym adresem [3].

Przypuszcza się, że właśnie w tym kierunku będzie zmierzał rozwój graficznych interfejsów użytkownika.

3 Materiały filmowe i teksty

Materiał filmowy oraz teksty są jednym z najważniejszym składników programu. Decydują one o przystępności aplikacji oraz jej przydatności. Najważniejszym celem przy przygotowywaniu części merytorycznej było upraszczanie. Nie robiliśmy bardzo szczegółowych opisów wymiany lub instalacji podzespołów komputera. Staraliśmy się pokazać to na filmach. Trudno o lepszy sposób przekazania wiedzy o tym na przykład jak bezpiecznie wymienić procesor.

Z każdym działem jest związany plik tekstowy zawierający informacje o danym podzespole komputera. Np. w dziale o procesorach można znaleźć informacje o tym jak, są produkowane, na co należy zwracać uwagę przy zakupie i jakie ich typy są obecnie najpopularniejsze na rynku.

3.1 Nagranie

Materiał nagrywany był kamerą Panasonic NVDS88 wyposażoną w przetwornik CCD 1/4" 800 tys. punktów. Kamera ta zapisuje materiał w standardzie DV na taśmie mini DV. Do nagrania tego materiału wykorzystano statyw firmy HAMA.

Od samego początku materiał nagrywany był bez dźwięku. Nagranie odbywało się przy świetle żarowym ustawionym tak, by zniwelować półcienie oraz poprawić kontrastowość obrazu. Wzorzec bieli kamery był także ustawiony na światło żarowe. Materiały umieszczone zostały na jednolitym tle.

Nagranie zawiera materiały filmowe oraz zdjęcia - nieruchome klatki. Całość materiału wynosi 90 minut filmowego o rozmiarze 720x576 pixeli.

Film został zgrany bezpośrednio do programu Adobe Premiere 6.02 za pomocą złącza FireWire (IEEE 1394) znajdującego się na karcie PCI wyposażonej w układ VIA 1394. Zgrywany materiał został zapisany z kompresją DV (30 klateksek. z opuszczeniem jednej klatki) w standardzie PAL. Jeden fragment nie był dłuższy niż 8 minut.

3.2 Montaż video

Montaż odbywał się za pomocą programu Adobe Premiere 6.02 - aplikacji służącej do nieliniowego montażu video oraz audio. Oferuje ona możliwość montażu bezkompresyjnego oraz pracy z materiałem skompresowanym. Program oferuje szereg efektów wizualnych np.: przejścia, transformacje obrazu oraz filtrowanie. Dostępne są także filtry audio oraz proste narzędzia służące do montażu dźwięku.

Pierwszym etapem montażu było wyekstraktowanie pojedynczych klatek do map bitowych zapisanych jako "tif". Format ten został wybrany ze względu na to, iż zapis w tym formacie nie pogarsza jakości zdjęć, co ważne jest przy dalszym montażu.

Kolejnym krokiem było pocięcie materiału na fragmenty tematyczne a następnie - sklejanie materiałów wykorzystując poniższe przejścia:
- Cube Spin - efekt przestrzennego obrócenia obrazu zastosowanego np. gdy pokazywano dwie różne

strony obiektu
- Flip Over - efekt przejścia symulującego ruch obrotowy
- Cross Dissolve - efekt wygaszania jednego obrazu oraz pojawiania się drugiego
- Additive Dissolve - efekt podniesienia luminacji pierwszego obrazu do całkowitej bieli oraz obniżeniu luminacji drugiego obrazu
- Push - efekt "przepchnięcia" jednego obrazu przez drugi
- Splash Side - efekt przysłonięcia jednego obrazu przez drugi w postaci zsuwających się fragmentów
- Cross Zoom - efekt przybliżenia fragmentu pierwszego obrazu a następnie pokazywaniu drugiego obrazu podczas oddalania pierwszego
- Zoom - efekt powiększania zaznaczonego fragmentu.
Wszystkie te efekty służą przestrzennemu zobrazowaniu wykonychwanych czynności, nadania obrazowi dynamicznego charakteru.
Obok przejść, zastosowano również filtry:
- Antialias - usuwanie ostrych krawędzi obiektów
- Camera View - wprowadzenie zmian w kącie widzenia danego obiektu (w naszym przypadku filtr wykorzystany do kadrowania pojedynczych sekwencji filmowych).

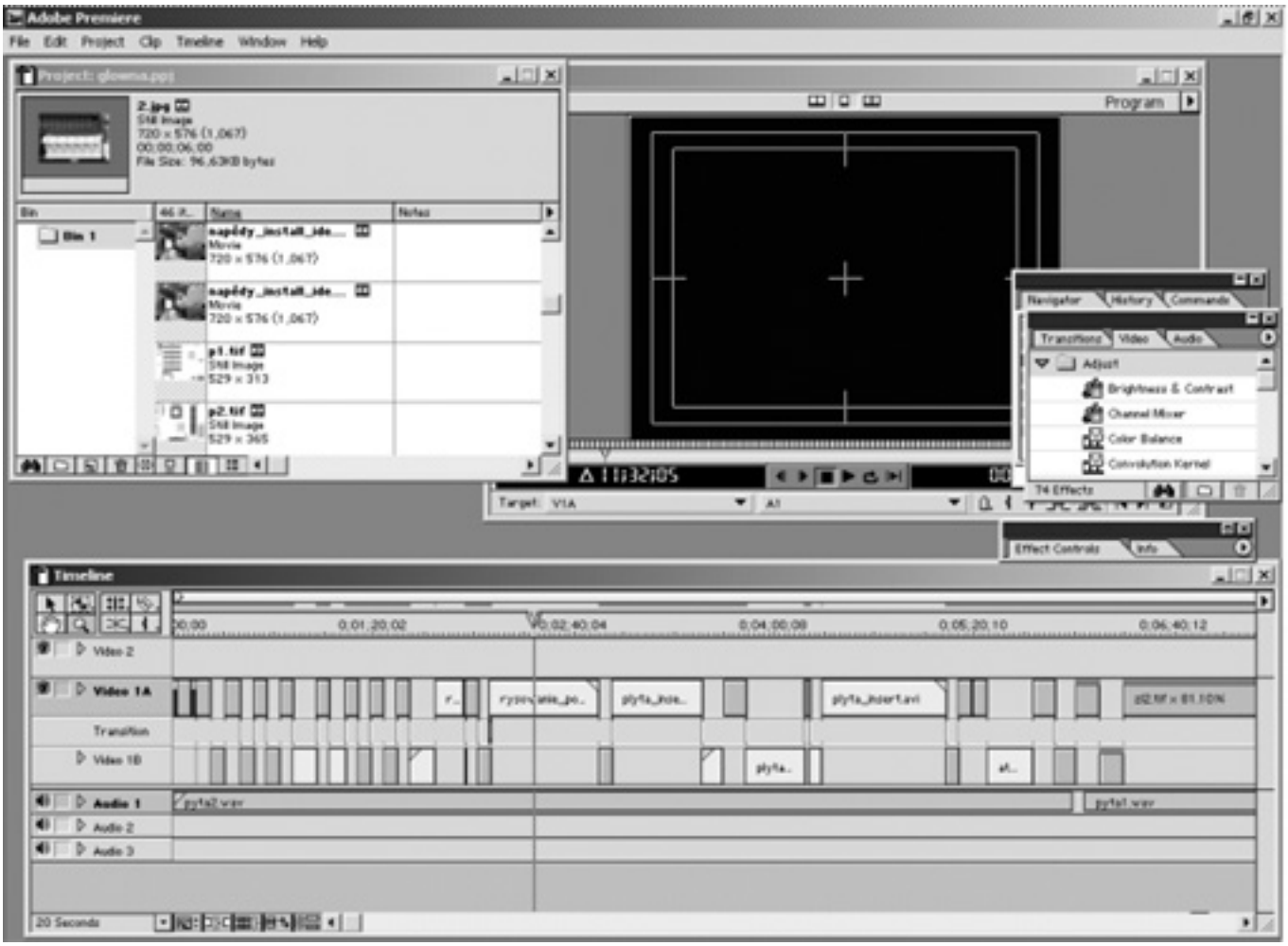
Do tak przygotowanego projektu, dodane zostały napisy oraz wstępny plik podglądu, który umożliwił nagranie lektora. Na rysunku 2 widać przykładową sekwencję wideo składającą się z 85 elementów.

3.3 Montaż audio

Dźwięk został nagrany za pomocą mikrofonu o czułości $-58db +/- 2$ wyposażonego w gąbkę tłumiącą. Dźwięk nagrywany był w postaci stereofonicznej z 16 bitową jakością oraz próbkowaniem 44100 Hz. Nagranie odbywało się przy pomocy programu Cool Edit Pro 2.0.

Plik zapisywany był bezpośrednio do formatu PCM - "Wav". Format ten został wybrany celowo, gdyż jest bezstratny. Tak przygotowany dźwięk był cięty na fragmenty a następnie sklejany. Otoczenie, w jakim dźwięk został nagrany, nie było bezgłośne. Poziom szumu od poziomu wartości sygnału wynosił ok. 45 db.. Taka wysoka różnica uniemożliwiała bezpośrednie zastosowanie filtrów odszumiających. Szum usunięto ręcznie, w pierwszej kolejności usuwając go z pauz pomiędzy wypowiadanymi przez lektora wyrazami a następnie z miejsc, gdzie był on słyszalny np. między połączeniami liter "k,i", "t,k", "k,u". Pewnym mankamentem tego typu filtracji jest nadmierna sterylność wypowiedzi, co powodowało jej nienaturalne brzmienie. Aby zminimalizować ten efekt, zastosowano filtr polegający na obniżeniu amplitudy dźwięku (Fade Out), aż do jego wygaszenia. Następnie, by usunąć pozostałe szumy, zastosowano filtr "Noise Reduction" na poziomie 40 db., przy małych współczynnikach korekcji. Kolejnym krokiem obróbki dźwięku było zastosowanie graficznego equalizer'a (Graphic Equalizer). Najpierw zastosowano ustawienie delikatnego usuwania wysokich tonów (Gentle Hi Cut) celem obniżenia tychże oraz pozbycia się innych szumów. Zastosowano również ustawienia delikatnego podnoszenia niskich tonów, by uzyskać "cieplejszy" ton głosu lektora. Tak przygotowany plik zapisywany był na dysku twardym.

Na rysunku 3 i 4 widzimy zdjęcia przedstawiające na jednym kanale dźwięk przed obróbką, drugi kanał przedstawia wersję finalną przetworzonego dźwięku.



Rysunek 2: Przykładowa sekwencja wideo

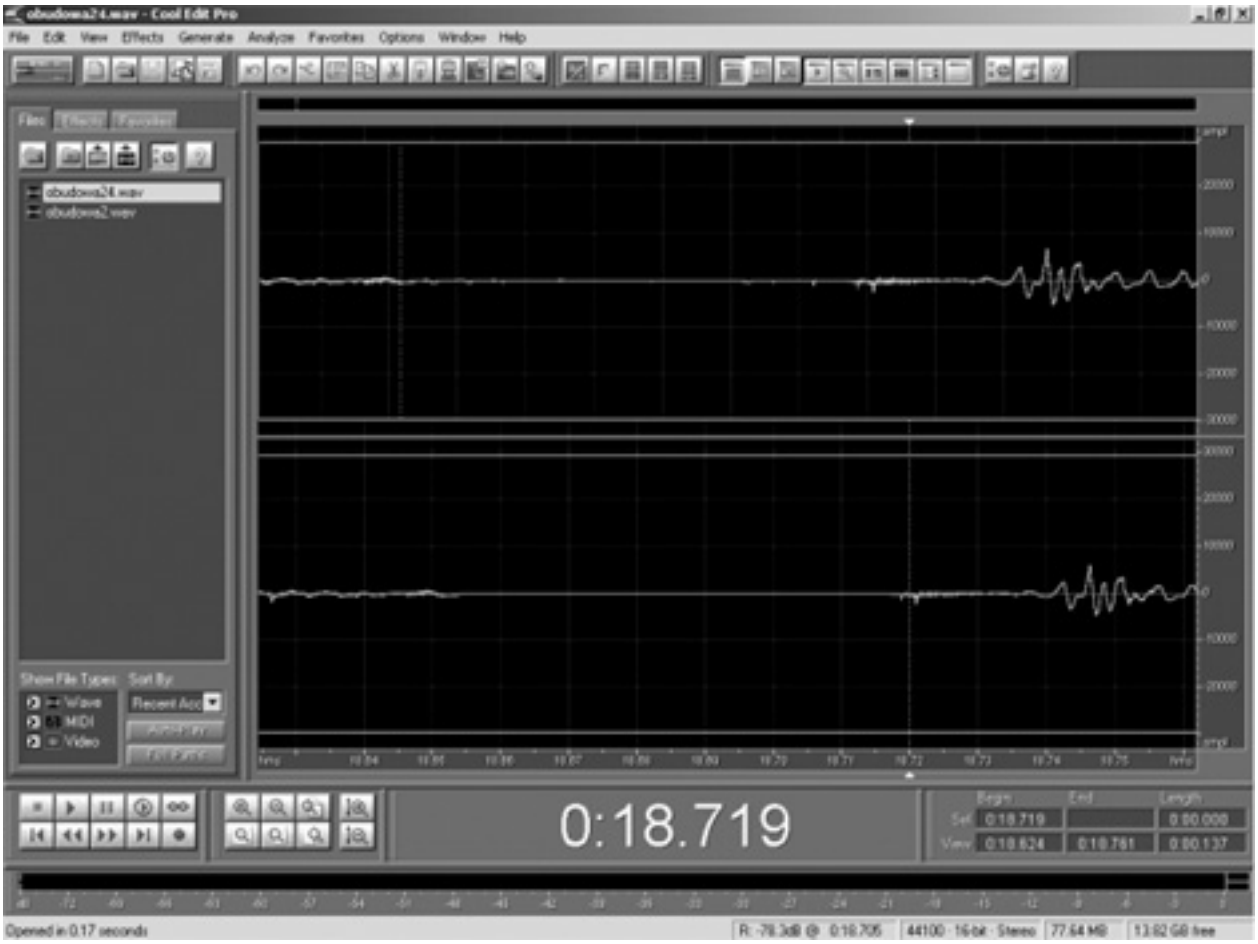
3.4 Efekty

Piki dźwiękowe "wav" przygotowane w programie Cool Edit Pro wklejane były do projektu video przygotowanego w programie Adobe Premiere. Materiał ten skompresowano przy zastosowaniu kompresji video - Divix oraz kompresji audio Microsoft ADPCM.

3.5 Zdjęcia

Istnieją cztery źródła zdjęć:
- kamera DV Panasonic NVDS88, rozmiar zdjęć 720x576 pixeli
- aparat Olymus CAMedia C2500L wyposażony w przetwornik CCD 4Mega Pixele (przesłona 7.8, czas naświetlania 1/125 sek.), rozmiar zdjęć 1712x1368 pixeli
- Internet - zdjęcia pochodzące z firmy Tom's Harware, rozmiar powyżej 800x600 pixeli
- skaner - Plustek Optic Pro UT 12 o rozdzielczości 600x1200 dpi.

Zdjęcia wykonane aparatem cyfrowym pochodzą ze studia fotograficznego, gdzie do ich zrobienia zastosowano dwie lampy błyskowe wyposażone w "soft box" - urządzenia zmiękczające i rozparszające światło błysku.
Zdjęcia wykonane kamerą cyfrową pochodzą z materiałów filmowych, o których mowa wcześniej.
Przy użyciu skanera wyposażonego w czujnik CCD skanowano obiekty przestrzenne przylegające



Rysunek 3:

do płyty skanera.
Obróbkę zdjęć wykorzystanych w dokumentach opisowych części oraz w poszczególnych filmach wykonano w programie Corel Photo-Paint 9.
Wszystkie strzałki i napisy oraz obrazy wymagające wektorowej obróbki sporządzono w programie Corel Draw 9. W tym programie wykonywano także skanowanie przedmiotów przestrzennych.
W niektórych przypadkach zdjęcia wymagały dodatkowych zabiegów - wyostrzania, kadrowania, montażu zdjęć.

4 Użyte formaty

Filmy wideo zostały skompresowane przy pomocy formatu divx. Główną zaletą tego formatu jest możliwość uzyskania dużego współczynnika kompresji przy zachowaniu wysokiej jakości obrazu. Zarówno kompresor jak i dekompresor można dowolnie konfigurować.

4.1 DivX

Format danych video - DivX spełnia taką samą funkcję przy zapisie obrazu, co format mp3 przy zapisie audio. Czasami nazywany jest także kodekiem,- jest to skrót od kodowania / dekodowania.

Rysunek 4:

Historia tego formatu jest dość burzliwa i nie brakuje w niej nawet wątków kryminalnych.

Nazwa wzięła się od komercyjnego produktu firmy Circuity City. Chodziło tu o filmy na płytkach CD i nowy system opłat za nie. Klient miał mieć możliwość wypożyczenia płytki z filmem, który mógł raz obejrzeć za darmo. Następne oglądanie miało być już opłacane. Specjalny odtwarzacz miał w tym celu tworzyć odpowiednią bazę danych o działaniach użytkownika. System ten jednak się nie przyjął, mimo że kosztowałby on kilkakrotnie mniej od filmów na płytkach DVD.

Pierwsza wersja DivX'a powstała na podstawie zhakowanego kodeka MPEG-4 który pierwotnie był stworzony przez Microsoft.

Kolejnym etapem w ewolucji "DivX ;-)" było stworzenie przez członków grupy Project Mayo kodeka OpenDivX na bazie standardu kompresji MPEG-4, a dokładniej na MoMuSys reference implementation of MPEG-4. Kod został napisany praktycznie od początku i jest obecnie dostępny dla każdego. Nie ma on nic wspólnego ze zhackowanym kodem Microsoft'u, tak więc jest zupełnie legalny.

Część II
Projekt od strony technicznej

Rozdział ten ma na celu odpowiedzieć na wszystkie techniczne pytania związane z projektem. Podzielony jest na trzy części. W pierwszej ma miejsce analiza systemu, czyli czego dotyczy, z czego się składa, jakich używa technologii. W drugiej części znajduje się szczegółowy opis systemu od strony technicznej. Część dokumentacji (dla kodu C++) została wygenerowana przy pomocy programu DOXYGEN.

5 O projekcie

Rysunek 5: Widoczna kontrolka WebBrowser wyświetlająca plik html

Ostatnio pojawiło się na rynku kilka ciekawych programów umożliwiających tworzenie aplikacji z dowolnie zaprojektowanym interfejsem użytkownika. Pierwszym z tych programów jest Flash firmy Macromedia, przeznaczony do tworzenia interaktywnych filmów umieszczanych na stronach internetowych. Drugim jest Director stworzony przez tą samą firmę, przeznaczony do tworzenia programów multimedialnych. Oba te programy mają swoje zalety i wady. Najważniejszą zaletą jest możliwości łatwego tworzenia interfejsu użytkownika. Zasadniczą wadą są liczne ograniczenia - Flash nie ma możliwość pokazywania w zadawalający sposób formatowanych dokumentów tekstowych. Podobnie pokazywanie filmów w takich formatach jak mpeg czy avi , jest mocno ograniczone. Director rozwiązuje wiele z tych problemów. Jest to jednak w pełni zintegrowany program, którego głównym przeznaczeniem są aplikacje multimedialne. Jeśli chce się stworzyć program wykorzystu-

jący jakieś urządzenia zewnętrzne albo inne programy (np. media player albo internet explorer), wówczas trzeba liczyć się z ograniczeniami.

Dlatego ciekawym pomysłem jest wykorzystanie ogromnych możliwości Microsoft Visual C++ do oprogramowania tych rozszerzeń, natomiast do stworzenia interfejsu wykorzystać można kontrolkę activex umożliwiającą pokazywanie filmików zrobionych we Flash'u. Do komunikacji między kontrolką a kodem napisanym w C++ używa się FS komand wysyłanych z języka skryptowego używanego we Flashu. Możliwa jest także komunikacja w drugą stronę.

Ponieważ głównym przeznaczeniem Flash'a jest tworzenie interaktywnych filmów umieszczanych na stronach internetowych, połączenie interfejsu zrobionego we Flash'u z kodem napisanym w c++ nie jest proste.

Wykorzystanie kontrolki Flasha do tworzenia interfejsu użytkownika przynosi wiele zalet. Jedną z nich jest oddzielenie pracy projektanta interfejsu użytkownika od pracy programisty aplikacji. Film zrobiony we Flashu może w pełni kontrolować działanie aplikacji. Przykładowo, jeśli ma być pokazana kontrolka WebBrowser oraz ma być w niej pokazany plik HTML projektant dodaje do klawisza akcje z następującym skryptem :

```
FSCommand("select html file","index.html"); FSCommand("show htmlview","");
```

W ten sposób nad kontrolką Flasha pojawi się kontrolka WebBrowser pokazująca plik index.html. Na początku skryptu trzeba jednak jeszcze zainicjalizować kontrolkę WebBrowser - tzn. podać jej położenie i rozmiary na ekranie. Dokonuje się tego poleceniem :

```
FSCommand("resize htmlview","10 10 500 400");
```

Spowoduje to umieszczenie lewego górnego rogu kontrolki WebBrowser na pozycji x=10 i y=10 względem lewego górnego rogu aplikacji. Kontrolka będzie miała także wymiary 500 pikseli na 400 pikseli. Twórca interfejsu użytkownika nie musi wcale wpisywać bezpośrednio liczb, może stworzyć Movie Clip (czyli obszar we Flashu, który ma określone położenie i wymiary) a potem pobrać jego położenie, szerokość i wysokość i wysłać je przez komendę "resize htmlview". Taki Movie Clip (pokazany na rysunku 4) można później dowolnie przemieszczać.

Powyższy przykład można zastosować do wielu innych kontrolek. W aplikacji "Składacz" wykorzystuje się taki sam mechanizm do pozycjonowania okna pokazującego pliki filmowe. Wszystkie te kontrolki mają zestaw FSCommand które sterują ich funkcjonalnościami.

5.1 Kontekst

Jest to multimedialny program do nauki rozbudowy komputera. Zalicza się on do programów edukacyjnych.

5.2 Cel pracy

Celem projektu jest uzyskanie aplikacji atrakcyjnej dla użytkownika pod względem estetycznym jak i merytorycznym. Ma ona mieć postać multimedialnej aplikacji edukacyjnej. Dodatkowym celem jest uzyskanie szkieletu aplikacji umożliwiającego łatwą rozbudowę i tworzenie nowych aplikacji.

Aplikacja opisywana w tym dokumencie ma na celu zaprezentować inne podejście do tworzenia ciekawych interfejsów użytkownika w aplikacjach napisanych w VC++. W metodzie tej używane są programy ogólnie dostępne na rynku (o niezbyt wygórowanych cenach). Umożliwa ona stworzenie ciekawych wizualnie programów, a także umożliwia oddzielenie pracy związanej z logiką programu od pracy związanej ze skomponowaniem interfejsu użytkownika.

Rysunek 6: Okno pliku swf z obszarem nad którym pokaże się kontrolka WebBrowser

5.3 Zastosowane technologie

Wysoka jakość interfejsu użytkownika jest uzyskiwana za pomocą programu Macromedia Flash. Umożliwia on komponowanie dowolnie wyglądających interfejsów użytkownika. Przy jego pomocy można nadawać kontrolkom (czyli przyciskom, polom edycyjnym itp.) dowolne kształty. Tworzone w ten sposób GUI[2] można wzbogacać o dynamiczne animacje.

Filmy są natomiast wyświetlane przy pomocy biblioteki Direct Show, dzięki czemu można wykorzystać windowsowe kodeki, a także akceleracje sprzętową. Główną zaletą tej biblioteki jest stosunkowa łatwość jej używania. Jest ona częścią pakietu Direct X, który jest obecnie w wersji 8 standardowo dodawany do nowych Windowsów.

Do prezentacji tekstu wykorzystana jest kontrolka Internet Explorer'a. Dzięki temu tekst może być napisany w HTML'u. Dodatkowo można wykorzystywać wszystkie te elementy, jakie są dostępne w Internet Explorerze, czyli np. aplety Javy albo Java Script czy nawet DHTML.

Aplikacja jest napisana z wykorzystaniem biblioteki MFC[3]. Szkielet aplikacji został stworzony przy pomocy AppWizard'a. Jest to program wchodzący w skład pakietu VC++ umożliwiający wygenerowanie szkieletu kodu. Biblioteka klas MFC jest najczęściej stosowana przy tworzeniu aplikacji pod systemem operacyjnym Windows. W bibliotece tej wszystkie kontrolki - czyli np. pola edycyjne, kontrolki activex albo przyciski są traktowane jako okna i wywodzą się z klasy CWnd. Klasa ta umożliwia wykonywanie podstawowych czynności na danym obiekcie, takich jak zmiana położenia, ukrywanie okna odkrywanie go.

Kształt aplikacji nie jest ograniczony do prostokątnego, może on być zmieniany z poziomu Flash'a (więcej w części technicznej dokumentacji).

Dodatkowo wykorzystano takie programy jak :

- **LaTeX**[4] - do stworzenia tej dokumentacji
- **DOXYGEN** - został użyty do stworzenia dokumentacji kodu źródłowego, z jego komentarzy

[2]Graphical User Interface - graficzny interfejs użytkownika
[3]Microsoft Foundation Classes
[4]z podprogramami - pdflatex,latex2html

- **Macromedia Flash 5** - posłużył do wykonania interfejsu użytkownika
- **Adobe Premiere 6.02** - z wykorzystaniem tego programu zmontowane zostało audio i video
- **Cool Edit Pro 2.0** - użyty został do stworzenia dźwiękowych efektów specjalnych

5.4 Zastosowane rozwiązania

Program ten jest mieszaniną różnych technologii. Aby mógł powstać, trzeba było sprawić, by mogły one ze sobą współpracować. Technologie takie jak Shockwave Flash, kontrolka WebBrowser[5], Direct Show udostępniają interfejsy, za pomocą których można je konfigurować i sterować nimi. Pod pojęciem interfejsu najczęściej rozumie się zestaw funkcji, przy pomocy których wydaje się obiektom różne polecenia. Praca naszego programu polega właśnie na konfiguracji tych elementów oraz na sterowaniu nimi.

Konfiguracja polega na określeniu początkowych ustawień oraz ewentualnie zmianie ich w czasie wykonywania programu. Na przykład kontrolka ShockWaveFlash może pokazywać pliki o rozszerzeniu swf zachowując niską, średnią albo wysoką jakość. Po uruchomieniu programu, aplikacja sprawdza, na jakim komputerze została uruchomiona i odpowiednio do możliwość sprzętu zmieniana jest jakość filmu Flasha. Konfiguracja kontrolki WebBrowser jest trochę bardziej skomplikowana. Aby mogła ona spełniać zadane wymagania, potrzebne było przedefiniowanie kilku klas.(więcej na ten temat w podrozdziale "O kontrolce WebBrowser")

Sterowanie natomiast polega na interakcji tych poszczególnych technologii ze sobą. Użytkownik może np. poprzez naciśnięcie przycisku wyrazić chęć przeczytania jakiegoś tekstu. Komunikat taki najpierw pojawi się w kontrolce ShockWaveFlash, gdzie uruchomi się odpowiednia procedura zarządzająca tym akurat zdarzeniem, następnie procedura ta przy pomocy funkcji fscommand wyśle do aplikacji - enginu - jaka strona ma się pokazać w przeglądarce. Engine natomiast tak ustawi przeglądarkę (kontrolkę WebBrowser), aby pokazywała tą właśnie stronę. Dodatkowo muszą być jeszcze zmienione ustawienia przewijaka pionowego w zależności od długości ładowanego tekstu.

System jest realizowany tak aby można było bez większych nakładów pracy, wykorzystać go przy innych projektach. Gotowy projekt składa się z enginu, którym jest aplikacja napisana w C++, interfejsu w postaci plików swf (rozszerzenie plików programu Flash),oraz plików multimedialnych (avi,mp3). Engine odpowiada za pokazywanie kontrolki ShockWaveFlash (która odgrywa filmiki zrobione we Flash'u), oraz za reagowanie na komendy wysyłane z niej. Przykładami komend są np.: odtwórz film, zminimalizuj aplikacje, zamknij aplikacje...

5.5 Zawartość

Część merytoryczna składa się ze zbioru plików tekstowych w formacie HTML oraz filmów w postaci plików z rozszerzeniem avi. Do plików HTML dołączone są także pliki graficzne w formacie jpeg. Pliki tekstowe można podzielić tematycznie:

1. Obudowa
2. Procesor
3. Pamięć
4. Płyta główna

[5]jest to kontrolka Internet Explorera

5. Napędy (w tym CD-ROM,DVD,FDD)

Interfejs użytkownika składa się z plików o rozszerzeniu swf[6] - są one tworzone w programie Flash firmy Macromedia wersja 5.0 . W skład całego interfejsu użytkownika wchodzą następujące pliki (nazwy plików można zmienić w cofig.ini):

- *frameview.swf* - plik ten zawiera ramkę okna aplikacji
- *videoframeview.swf* - jest to interfejs okienka dialogowego pokazującego filmy wideo

W skład programu wchodzi także plik o rozszerzeniu rgn[7]. Przechowuje on dane dotyczące kształtu aplikacji. Może on być zdefiniowany dla każdego okna aplikacji. Plik regionu jest generowany przy pomocy programu Region, który jest dokładnie opisany w rozdziale "Program Region".

Konfiguracja aplikacji znajduje się w pliku config.ini. W pliku tym można określić, w jakich katalogach program ma szukać poszczególnych mediów. Można również określić w nim początkowy rozmiar aplikacji, nazwę, to czy ma być skalowana przez użytkownika. W rozdziale Konfiguracja znajduje się pełny opis zmiennych pliku konfiguracyjnego.

Znaczenie katalogów na płytce:

. - w katalogu głównym znajdują się pliki potrzebne do instalacji programu "Składacz"
video - w tym katalogu znajdują się pliki wideo
flash - instalator Macromedia Flash Player
projekt/program region - Program Region z przykładowym interfejsem użytkownika
projekt/region_doc - dokumentacja w postaci html programu region, uruchamia się ją przez index.html
projekt/skladacz_doc - dokumentacja w postaci html programu Składacz, uruchamia się ją przez index.html
projekt/zrodla - kod źródłowy
projekt/lektorka - znajdują się tu skrypty użyte podczas nagrywania głosu lektora

6 Dokumentacja techniczna

Rozdział ten zawiera opis aplikacji oraz zastosowanych w niej rozwiązań.

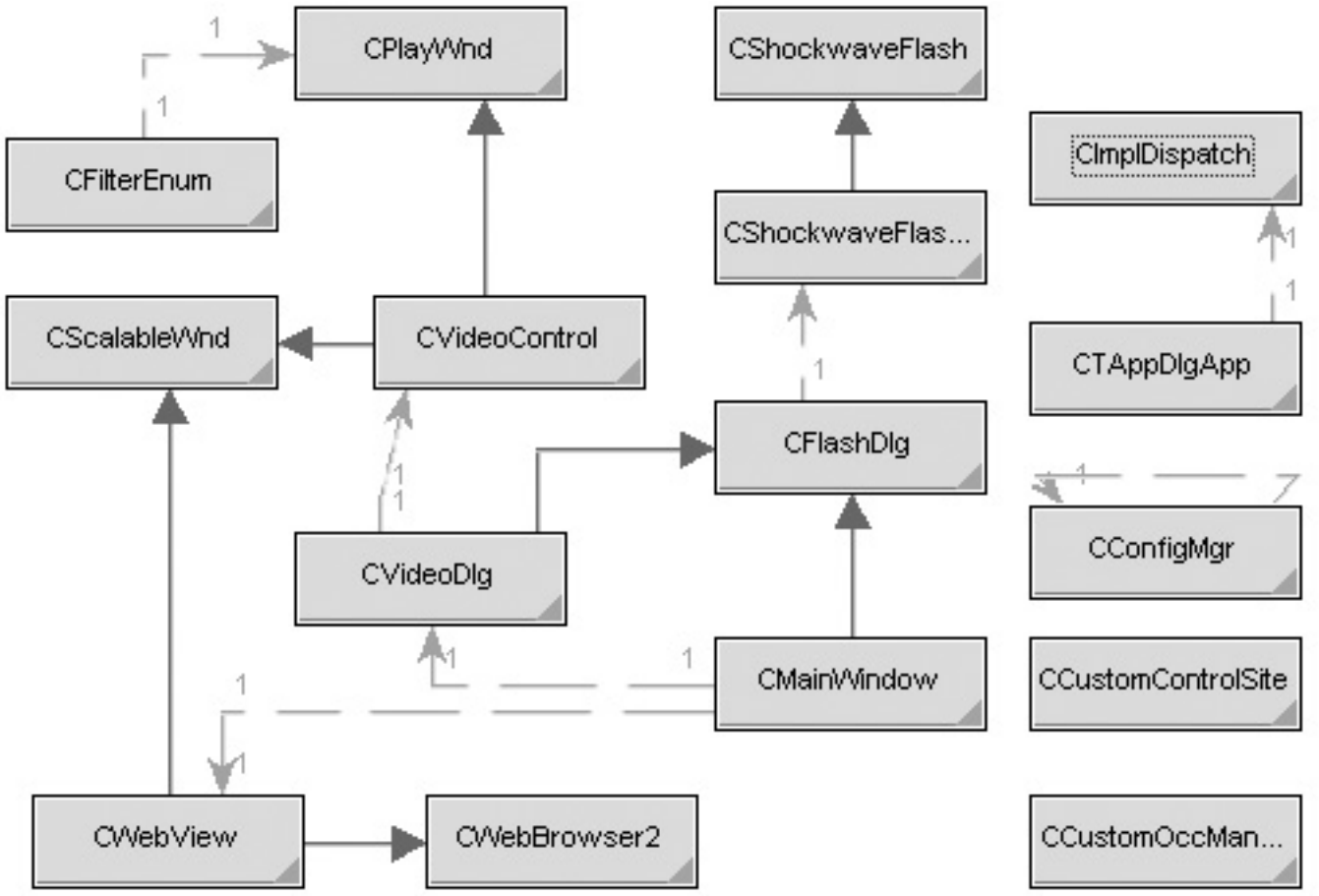
Szczegółowy opis wszystkich klas programu znajduje się w dokumentacji wygenerowanej programem DOXYGEN i można ją znaleść na załączonej płytce. Program ten generuje dokumentację na podstawie komentarzy wstawianych w kodzie źródłowym. Na końcu tego dokumentu znajdują się wydruki plików nagłówkowych i implementacyjnych najważniejszych klas programu.

W ostatnim podrozdziale znajduje się opis programu Region, który służy do generowania plików rgn, wykorzystywanych do zmiany kształtu aplikacji.

6.1 VC++

Aplikacja została napisana w języku C++ z wykorzystaniem środowiska programistycznego Microsoft Visual C++. Opiera się o szkielet aplikacji dialogowej, który został wygenerowany przy pomocy wewnętrznego kreatora MSVC[8].

[6] shockwave flash
[7] skrót od region
[8] Microsoft Visual C



Rysunek 7: Diagram klas głównego programu. Strzałki ciągłe oznaczają z której klasy dana klasa dziedziczy. Linie przerywane oznaczają, że klasa zawiera zmienne składowe wskazywanej klasy. Na diagramie nie jest to pokazane, ale klasy CShockWaveFlash i CWebBrowser2 dziedziczą z klasy CWnd - która jest klasą należącą do biblioteki MFC. Klasa CFlashDlg dziedziczy natomiast z CDialog, która także jest częścią MFC.

Na rysunku 5 widać diagram klas. Pierwszą funkcją, która jest wykonywana po uruchomieniu programu jest metoda

```
BOOL CTAppDlgApp::InitInstance()
```

z klasy CTAppDlgApp. W metodzie tej odczytywana jest najpierw konfiguracja z pliku config.ini. Zadanie to wykonuje obiekt klasy CConfigMgr.Jest to singleton, co oznacza że może istnieć tylko jedna instancja tej klasy. Takie podejście zapewnia, że nie zostanie utworzona druga instancja tej klasy. Do obiektu tej klasy można odwoływać się przy pomocy CConfigMgr::Get(), co zwraca referencję lub krócej appconfig() co również zwraca referencję

Funkcja InitInstance() tworzy główne okno aplikacji : CMainWindow. Uruchamiane jest w trybie modalnym, co oznacza że funkcja DoModal() wywołana na obiekcie CMainWindow zakończy swoje działanie dopiero po zamknięciu okna. Co następuje po wywołaniu (wewnatrz CMainWindow) funkcji CDialog::OnOK().

Klasa CMainWindow dziedziczy z klasy CFlashDlg. Klasa CFlashDlg tworzy kontrolkę[9] ac-

[9]termin ten określa takie obiekty jak przyciski, suwaki, okienka tekstowe itp.

tivex[10] o nazwie shockwaveflash, która pokazuje na całej powierzchni okna film swf zrobiony w programie Macromedia Flash. CFlashDlg odbiera komunikaty (fscommand) z kontrolki activex i odpowiednio na nie reaguje. Komunikaty te są wysyłane przez kontrolkę flasha w odpowiedzi na różne działania użytkownika (np. kliknięcia na przyciskach).

W metodzie OnInitDialog klasy CMainWindow, tworzone jest okienko dialogowe klasy CVideoDlg, które będzie pokazywać filmy wideo. Początkowo jest ono ukryte. Klasa CVideoDlg zawiera w sobie zmienną typu CVideControl, jest ona odpowiedzialna za renderowanie filmów. Do odtwarzania plików filmowych wykorzystywana jest biblioteka Direct Show z pakietu DirectX. CPlayWnd jest klasą która umożliwia dostęp do tej biblioteki.

Poniżej znajduje się krótki opis poszczególnych klas. Bardziej szczegółowe opisy znajdują się w dokumentacji wygenerowanej programem DOXYGEN oraz w postaci komentarzy w plikach źródłowych.

CCustomControlSite - służy do ukrycia menu kontekstowego,
CCustomOccManager - zarządza klasą CCustomControlSite
CImpIDispatch - służy do rozszerzania możliwości kontrolki WebBrowser'a

CWebBrowser2 - klasa kontrolki pokazującej pliki HTML
CShockwaveFlash - klasa kontrolki ShockWave

CConfigMgr - klasa zarządza plikiem konfiguracyjnym
CFilterEnum - otwiera okno filtrów
CFlashDlg - zarządza kontrolką flasha i regionami
CMainWindow - główne okno aplikacji
CPlayWnd - odgrywa filmy w oknie
CScalableWnd - służy do skalowania kontrolek
CShockwaveFlashAddOn - kontroluje zachowanie kontrolki Flasha
CTAppDlgApp - główna klasa aplikacji gdzie wszystko się zaczyna
CVideoControl - klasa sterująca wyświetlaniem filmami wideo
CVideoDlg - klasa okna dialogowego pokazującego film
CWebView - klasa ta steruje kontrolką WebBrowser

Klasy CShockwaveFlash i CWebBrowser2 są automatycznie generowane przez Microsoft Visual C++ przy dodawaniu nowej kontrolki lub komponentu do projektu[11]. Przy pomocy takiej klasy można dynamicznie tworzyć nowe obiekty kontrolek w aplikacji. Klasy CCustomControlSite, CCustomOccManager, CImpIDispatch zostały dodane do projektu zgodnie z artykułem Q236312 znajdujacym sie w Microsoft Knowledge Base (więcej na ten temat w rozdziale 'O kontrolce WebBrowser').

6.1.1 Dowolny kształt aplikacji

Dowolny kształt aplikacji jest realizowany przy pomocy standardowych funkcji winapi. Funkcja SetWindowRgn() przekazuje oknu strukturę danych zawierającą informacje o żądanym wyglądzie aplikacji. Struktura ta jest to tablica zawierająca obiekty typu RECT.

[10] kontrolki activex to takie kontrolki które mogą być np. osadzane w plikach html

[11] W MSVC nowe kontrolki dodaje się wybierając z menu Project, Add To Project a następnie Components and Controls

Rysunek 8: Okienko wyświetlające wideo

```
typedef struct _RECT {
    LONG left;
    LONG top;
    LONG right;
    LONG bottom;
} RECT;
```

Każdy obiekt typu RECT zawiera informacje o współrzędnych położenia prostokątu oraz jego rozmiarach. Także region aplikacji tak naprawdę składa się z dużej ilości prostokątów. Jeśli linia granicy aplikacji jest mocno zakrzywiona, wówczas potrzeba większej ilości prostokątów do określenia takiego regionu, co negatywnie wpływa na odświeżanie wszystkiego co znajduje się pod spodem aplikacji.

6.1.2 O kontrolce *WebBrowser*

Jest to ta sama kontrolka jakiej używa Internet Explorer do pokazywania plików HTML. W programie "Składacz" jest ona pozbawiona ramki oraz menu kontekstowego, które normalnie pojawiało by się po kliknięciu prawym klawiszem mysz na kontrolce. Jest to możliwe dzięki trzem klasom implementującym nową funkcjonalność kontrolki WebBrowser2 : CImplDispatch, CCustomControlSite, CCustomOccManager. Ta zmiana funkcjonalności kontrolki WebBrowser2 została zrobiona zgodnie z procedurą opisaną w artykule Q236312 znajdującym się w Microsoft Knowledge Base[12].

[12] `http://support.microsoft.com/default.aspx?scid=KB;EN-US;Q236312&`

6.1.3 O bibliotece Direct Show

Direct Show jest częścią biblioteki DirectX. DirectX umożliwia programistom tworzenie wydajnych aplikacji multimedialnych. Dzieli się na takie biblioteki jak Direct3D - służące do tworzenia aplikacji 3D, Direct Play - umożliwiającą grę w gry komputerowe przez internet, Direct Music i Direct Sound służące do odgrywania i nagrywania dźwięków.

Direct Show ma za zadanie odtwarzać pliki multimedialne o różnych formatach. Mogą to być zarówno pliki audio jaki i wideo. W programie "Składacz" odtwarzane są tylko filmy wideo. W poprzedniej wersji tego programu do odtwarzania filmów wykorzystywana była biblioteka MCI[13]. Udostępnia ona wysokiego poziomu interfejs służący do odgrywania plików multimedialnych. Na przykład, aby odegrać w MCI film "pearljam.mp3" wystarczyłoby wysłać przy pomocy funkcji mciSendString() ciąg "play pearljam.mp3". MCI ma jednak wady. Największą z nich jest niezdolność do poprawnego odgrywania popularnych plików wideo takich jak DivX. Direct Show nie ma tych ograniczeń.

Za tworzenie i zarządzanie filmami odpowiada klasa CPlayWnd. Została ona oparta na przykładzie z DirectX o nazwie playwnd. Klasa ma za zadanie otworzyć plik i pobrać wszystkie niezbędne interfejsy pozwalające spełniać kontrolę nad tym plikiem. Interfejsy oznaczają pewne funkcjonalności np.: IMediaSeeking umożliwia zmianę aktualnej pozycji wyświetlanego filmu. Interfejs IMediaEventEx umożliwia odbieranie różnych komunikatów związanych ze stanem wyświetlanego filmu. Komunikatem może być np. zakończenie się filmu.

Klasa CPlayWnd ukrywa fakt tego, że do renderowania filmów wykorzystuje się Direct Show. Dzięki temu z minimalnym nakładem pracy można zmienić bibliotekę Direct Show na jakąś inna.

6.1.4 Plik config.ini

Plik ten zawiera konfigurację aplikacji. Jest on potrzebny, aby móc swobodnie dokonywać zmian w programie bez potrzeby rekompilowania go. Dodatkowo jest on niezbędny osobie tworzącej interfejs użytkownika, gdyż osoba ta najczęściej nie ma możliwości dokonania rekompilacji systemu. W pliku można używać komentarzy, które zaczynają się od znaku # i sprawiają, że wszystko po prawej stronie znaku jest pomijane. Zmienne które nie mają podanej wartości muszą mieć po prawej stronie znak równości. Plikiem konfiguracyjnym zarządza klasa CConfigMgr. Ponizej znajduje się opis zmiennych pliku config.ini[14] :

ADD_MODULE_PATH= dodaje ścieżkę prowadzącą do aplikacji, do plików swf i rgn
FRAME_SWF_FILENAME= określa nazwę pliku zawierającego główny interfejs aplikacji
RGN_FILENAME= nazwa pliku rgn dla głównego okna aplikacji
VIDEO_SWF_FILENAME= nazwa pliku swf dla okna pokazującego filmy
RGN_VIDEO_FILENAME= nazwa pliku rgn dla okna pokazującego filmy
VIDEO_DIRECTORY_PATH= katalog zawierający filmy wideo , zmienna ta nie jest używana jeśli zdefiniowana jest zmienna LOCALIZE_CD_VIDEO_PATH=.
CD_NAME= nazwa płytki zawierającej filmy wideo (np.: skl)
LOCALIZE_CD_VIDEO_PATH= pliku wideo będą szukane na płytce CD w podanym katalogu
HTML_DIRECTORY_PATH= nazwa katalogu zawierającego pliku html
MAINWINDOWMODE= tryb pracy okna głównego (standardowo : FRAMEVIEWSWF)
APPLICATION_NAME= nazwa aplikacji (np.: Składacz)
VIDEO_WINDOW_INITIAL_SCALE= poczatkow skala okna pokazujacego filmy, szerokość i wysokość sa skalowane równomiernie (np.: 0.5)

[13]Media Control Interface
[14]dodatkowe opisy można znaleść w config.ini aplikacji Składacz

HIDE_BROWSER_WHILE_RESIZING= ukryj przeglądarkę html, na czas zmiany rozmiarów aplikacji

SWF_QUALITY_WHILE_RESIZING= jakość pliku swf w czasie zmiany rozmiarów (2-wysoka 1-średnia 0-niska)

ONE_TO_ONE_SCALING= skalowanie wszystkich okien proporcjonalnie do ich wielkości początkowych

NO_MAIN_WINDOW_SCALING= główne okno aplikacji nie ma zmieniać rozmiarów

MAIN_WINDOW_INITIAL_SCALE= początkowe skalowanie aplikacji (np.: 1.0 1.0)

FLASH_VIDEO_FIELD_POS= ścieżka dostępu do zmiennej (w pliku swf) określającej pole na którym można klikać aby zmienić pozycję filmu (np.: _root.MoviePos.video_field_pos)

FLASH_VIDEO_POSITION= ścieżka dostępu do zmiennej (we pliku swf) pokzującej znacznik pozycji filmu (np.: _root.MoviePos.video_position)

FLASH_MAIN_WINDOW_WIDTH= szerokość głównego okna aplikacji

FLASH_MAIN_WINDOW_HEIGHT= wysokość głównego okna aplikacji

FLASH_VIDEO_WINDOW_WIDTH= szerokość okna dialogowego pokazujacego film

FLASH_VIDEO_WINDOW_HEIGHT= wysokość okna dialogowego pokazujacego film

STOP_MAINWINDOW_FLASH_WHILE_VIDEO= zamraża kontrolkę shockwaveflash w czasie gdy pokazywany jest film, może to zwiększyć wydajność aplikacji na słabszych komputerach.

STOP_FLASH_TIMERS_WHILE_RESIZING= zamraża kontrolkę shockwaveflash w czasie zmiany rozmiaru aplikacji.

TRANSLATE_VIDEO_POSITION= przesuwa okno nad którym wyświtlany jest film o kilka pikseli. Wartością może być np.: 1 3 - co przesunie okno o 1 piksel na osi x i 3 piksele na osi Y.

SWF_HIGH_QUALITY= określa od jakiej częstotliwości procesora flash ma mieć wysoką jakość (np.700)

SWF_MEDIUM_QUALITY= określa od jakiej częstotliwości procesora flash ma mieć wysoką medium (np.450)

DONT_LOG_TO_PROGRAM_LOG= wyłącza logowanie do pliku program.log

Plik config.ini musi znajdować się głównym katalogu aplikacji.

6.2 Flash

W pierwszej wersji tego projektu interfejs użytkownika był podzielony na dwa pliki swf. Pierwszy reprezentował ramkę aplikacji - czyli przyciski minimalizacji i zamykania programu, a drugi pokazywał wnętrze aplikacji. Ten drugi plik był pokazywany przez plik ramki na odrebnej warstwie w postaci "movie clipu". Okazało się jednak, że tworzenie interfejsu w taki sposób może być dość kłopotliwe. Trudno jest zrobić gładkie przejścia między oboma plikami swf, gdyż we Flashu trzeba je edytować osobno.

Lepszym rozwiązaniem jest stworzenie jednego pliku swf dla całego interfejsu użytkownika. Jeden plik swf zawiera teraz takie elementy jak górną belkę okna (służącą do przesuwania aplikacji) z przyciskami przejścia do pełnego ekranu, minimalizacji oraz zamknięcia aplikacji. Poza tym w prawym dolnym rogu znajduje się obszar służący do zmiany rozmiaru aplikacji.

6.2.1 Action script

Action script jest to wewnętrzny język programu Macromedia Flash. Jest on najczęściej wykonywany w odpowiedzi na różne działania użytkownika. Na przykład naciśnięcie przycisku powoduje uruchomienie pewnego fragmentu skryptu - ten fragment kodu w terminologii programu Flash nazywa się akcją. Sam Action Skrypt został stworzony na podstawie standardu Java Skryptu, jakkolwiek nie wymaga on deklarowania zmiennych, ani nie dokonuje ścisłej kontroli typów (The European

Computers Manufacturers Association (ECMA) stworzyła dokument nazywany ECMA-262[15] będący międzynarodowym standardem języka JavaScript, ActionScript został oparty właśnie na tym dokumencie).

W tym rozdziale opisany jest sposób, w jaki film zrobiony we Flashu komunikuje się z aplikacją. Komunikacja ta odbywa się za pomocą komendy FSCommand(String komenda , String argument). Jest ona podczepiana jako akcja pod różne elementy interfejsu. Wywołanie tej funkcji w filmiku flasha powoduje wystąpienie zdarzenia, które jest odbierane przez metodę : OnFSCommandShockwave (klasy CFlashDlg).

Spis wszystkich FSCommand znajduje się w podrozdziale 6.2.2. . Większość z nich jest na tyle prosta, że może być wywoływana samodzielnie. Na przykład fscommand("exit") po prostu zamyka aplikację. Istnieje jednak grupa fscommand, które muszą być wywoływana w określonej kolejności.

Poniższy kod jest przykładowym skryptem inicjalizującym okno html:

```
function init_html_view() {
    xs = (_root.Html_view.html_view_pos._xscale/100.0);
    ys = (_root.Html_view.html_view_pos._yscale/100.0);

    tst =  _root.Html_view.html_view_pos._x + " ";
    tst += _root.Html_view.html_view_pos._y + " ";
    tst += (_root.Html_view.html_view_pos._width*xs )+" ";
    tst += (_root.Html_view.html_view_pos._height*ys );

    fscommand ("resize htmlview", tst);             //1
    fscommand ("select html file", "index.htm");    //2
    fscommand ("show htmlview");                    //3
}
```

Na końcu funkcji wywoływane są trzy fscommandy, pierwsza powoduje zmianę wymiarów okna pokazującego plik html. Zgodnie ze specyfikacją z rozdziału 6.2.2. argumentem tego wywołania muszą być cztery liczby całkowite, określające umiejscowienie okna oraz jego szerokość i wysokość. Liczby te są obliczane na początku funkcji. Rysunek 4 pokazuje prostokątny obszar, na którym będzie pokazywana kontrolka internet explorera. W terminologii przyjętej w programie flash, nazywa się on Move Clipem . Takim movie clip'om można nadawać nazwy. W tym przykładzie nazywa się on _root.Html_view.html_view_pos (wraz ze ścieżką dostępu). Posiada on takie atrybuty jak _xscale i _yscale - określające jak bardzo obszar ten został przeskalowany, _x i _y określające jego pozycję na ekranie oraz _width i _height będące jego szerokością i wysokością. Po obliczeniu pozycji x,y oraz wymiarów , są one składane do stringa który jest później wysyłany do aplikacji.

Druga fscommenda wybiera plik html, który ma być pokazywany a trzecia powoduje pokazanie się przeglądarki html. Nie będzie ona posiadać ramek, więc doskonale wkomponuje się w otoczenie. Dodatkowo można łatwo określić w plikach html kolor tła taki sam jak kolor pliku swf w bliskim sąsiedztwie kontrolki WebBrowser, dzięki czemu nie będzie widać granicy między interfejsem zrobionym we flashu a kontrolką internet explorera pokazującą plik html.

6.2.2 Spis *FScommand*

Poniżej znajduje się spis komend - czyli parametrów funkcji fscommand - jakie można wysłać z poziomu skryptu w filmiku zrobionym we flashu. Wywołanie funkcji wymaga podania dwóch para-

[15] http://www.ecma.ch/

metrów, tzn. komendy i argumentu. Czasami wymagana jest tylko komenda. Niektóre parametry składają się z liczb rzeczywistych, w takim wypadku liczba rzeczywista będzie poniżej reprezentowana jako float, łańcuchy będą natomiast reprezentowane poprzez string. Liczby całkowite są reprezentowane jako int. Nic się nie stanie, jeśli wyśle się liczbę rzeczywistą zamiast całkowitej, w takim przypadki program zaokrągli ją. Znak minus " - " oznacza, że dla danej komendy argument nie jest wymagany i w funkcji fscommand() można pominąć ten parametr.

Niektóre komendy wymagają, aby ich wywołanie było poprzedzone wywołaniem innej komendy - inicjalizujacej jakąś funkcjonalność.

Wywołanie komendy polega na wstawieniu w kodzie źródłowym wywołania funkcji FScommand, np:

```
fscommand ("play video","pearljam.mpg");
```

powoduje wyświetlenie filmu z pliku o nazwie pearljam.mpg. Ta komenda musi być poprzedzona wywołaniem komendy "resize video", która to inicjalizuje rozmiary okna wideo. Funkcja fscommand składa się z dwóch parametrów : komendy i argumentu. Niektóre komendy wymagają, aby argument składał się z szeregu różnych pod-argumentów, należy je wówczas oddzielić od siebie spacjami.

Poniższe komendy są wspólne dla wszystkich rodzajów okien

Komenda:
fullscreen
Argument:
-
Opis:
Jeśli zostanie wysłana podczas, gdy aplikacja znajduje się w trybie okienkowym, wówczas aplikacja przechodzi do trybu pełnoekranowego. Jeśli natomiast aplikacja znajduje się w trybie pełnoekranowym wówczas wysłanie tej komendy powoduje przejście z powrotem do trybu okienkowego.

Komenda:
resiz
Argument:
true
Opis:
Wysłanie tej komendy sprawia, że aplikacja przechodzi do stanu zmieniania rozmiaru. Stan ten będzie się utrzymywał dopóki użytkownik będzie trzymał wciśnięty przycisk myszy. Dlatego też komenda ta powinna być powiązana z obszarem (we Flashu), na którym użytkownik może nacisnąć przycisk myszy. Naciśnięcie przycisku myszy na tym obszarze powoduje wysłanie tej komendy. Teraz gdy użytkownik będzie poruszał myszą (trzymając wciśnięty przycisk myszy) aplikacja będzie odpowiednio zmieniać swoje rozmiary.

Obszar wysyłający tą komendę najlepiej jest umieścić w prawym dolnym rogu interfejsu użytkownika. A obszar powinien być typu symbolem typu button.

Komenda:

Window Can Move

Argument:

-

Opis:

Sprawia. że aplikacja przechodzi do stanu zmiany położenia okna. Po wysłaniu tej komendy użytkownik może (trzymając wciśnięty przycisk myszy) przesuwać okno. Ze stanu tego aplikacja wyjdzie w momencie gdy użytkownik puści przycisk myszy.

Obszar, który ma wysyłać tą komendę najlepiej umieści w postaci belki umieszczonej w górnej części ekranu. (we flashu: należy wybrać z menu Insert, Convert To Symbol) Belka ta powinna być skonwertowana do symbolu typu button. A fscommand'a "Window Can Move" powinna być wysłana w odpowiedzi na zdarzenie "on press" myszki :

```
on (press) {
    fscommand ("Window Can Move");
}
```

.

Komenda:

exit

Argument:

-

Opis:

Powoduje wyjście z aplikacji.

Komenda:

minimize

Argument:

-

Opis:

Minimalizuje aplikacje.

Poniższe komendy są wykorzystywane w okienku wyświetlającym kontrolkę WebBrowser

Komenda:

resize htmlview

Argument:

float float float float

Opis:

Komenda ta powoduje określenie położenia oraz rozmiarów okienka kontrolki WebBrowser. Pierwsze dwie liczby rzeczywiste określają współrzędną (x,y), lewego górnego rogu kontrolki. Natomiast ostatnie dwie liczby rzeczywiste określają szerokość oraz wysokość kontrolki. Współrzędne są określane względem górnego lewego rogu aplikacji.

Komenda ta jest najczęściej wysyłana w sekcji inicjalizacyjnej action skryptu. Współrzędne najlepiej jest pobrać z prostokątnego obszaru będącego Movie Clipem. We flashu można go dowolnie później pozycjonować. Obszar ten będzie później zasłonięty przez kontrolkę WebBrowser.

Komenda:
select html file
Argument:
string
Opis:
Powoduje załadowanie nowego pliku html do przeglądarki - kontrolki WebBrowser. Na przykład:

```
fscommand ("select html file", "index.htm");
```

powoduje otwarcie pliku index.htm , ścieżkę dostępu do pliku html określa się w pliku config.ini za pomocą zmiennej : HTML_DIRECTORY_PATH=.

Komenda:
scroll down down
Argument:
float
Opis:
Powoduje, że kontrolka WebBrowser rozpocznie przewijanie zawartości pokazywanego pliku html do dołu. Argumentem tu jest wartość, o jaką ma być przesunięty ekran w każdej sekundzie kiedy użytkownik trzyma wciśnięty przycisk myszy, na danym przycisku. Program składacz wykorzystuje tu wartość 15 jako argument.

Komenda:
scroll down up
Argument:
-
Opis:
Sygnalizuje aplikacji, że użytkownik zwolnił przycisk przewijania i że aplikacja powinna przestać przewijać tekst w dół.

Komenda:
scroll up down
Argument:
float
Opis:
Powoduje, że kontrolka WebBrowser rozpocznie przewijanie zawartości pokazywanego pliku html do góry. Argumentem tu jest wartość, o jaką ma być przesunięty ekran w każdej sekundzie kiedy użytkownik trzyma wciśnięty przycisk myszy, na danym przycisku. Program składacz wykorzystuje tu wartość 15 jako argument.

Komenda:
scroll up up
Argument:
-
Opis:
Sygnalizuje aplikacji, że użytkownik zwolnił przycisk przewijania i że aplikacja powinna przestać przewijać tekst w górę.

Komenda:
show htmlview
Argument:
-
Opis:
Powoduje pojawienie się kontrolki WebBrowser. Powinna być wysłana przed rozpoczęciem przeglądania pliku html.

Komenda:
scroller height
Argument:
float
Opis:
Służy do określenia wysokości suwaka. Wartość ta powinna być równa odległości pomiędzy dwoma punktami granicznymi w jakich może się znaleźć suwak.

Poniższe komendy są wykorzystywane w okienku wyświetlającym film

Komenda:
exit
Argument:
-
Opis:
Poza tym że zamyka okno, zatrzymuje także wyświetlany film.

Komenda:
resize video
Argument:
float float float float
Opis:
Rozszerza okno wyświetlające film. Dwie pierwsze liczby rzeczywiste oznaczają współrzędną x i y górnego lewego rogu okna, natomiast dwie ostatnie współrzędne określają jego szerokość oraz wysokość.

Komenda:
select video name
Argument:
string
Opis:
Umożliwia ustawienie nazwy pliku filmu, który ma być wyświetlany. Standardowo argument powinien zawierać tylko nazwę pliku z rozszerzeniem, plik natomiast powinien znajdować się w katalogu ”..
data
” , względem katalogu w którym znajduje się aplikacja.

Komenda:
play video
Argument:
string
Opis:
Zaczyna wyświetlanie pliku video, podanego w argumencie.

Komenda:
play video part
Argument:
string int int
Opis:
Zaczyna wyświetlanie pliku video, o podanej nazwie oraz sprawia, że okno będzie wyświetlać ten film tylko w określonym przedziale. Pierwszy int oznacza sekundę, w której film ma mieć swój początek a drugi int określa sekundę, w której film ma mieć swój koniec.

Komenda:
start video
Argument:
-
Opis:
Zaczyna wyświetlanie pliku video.

Komenda:
stop video
Argument:
-
Opis:
Kończy wyświetlanie pliku video oraz przewija go do początku.

Komenda:
pause video
Argument:
-
Opis:
Zatrzymuje wideo w miejscu.

Komenda:
increase volume
Argument:
-
Opis:
Zwiększa głośność (o 10dB).

Komenda:
decrease volume
Argument:
-
Opis:
Zmniejsza głośność (o 10dB).

Komenda:
move video pos
Argument:
float
Opis:
Powinno być wywoływane, gdy użytkownik kliknie na obszarze zmiany pozycji filmu, argumentem jest wartość z przedziału [0.0,1.0] i określa procentowo nową pozycję filmu.

6.3 Visual Basic

Visual Basic 6.0 jest pakietem oprogramowania wyprodukowanym przez firmę Microsoft. Jest środowiskiem tworzenia aplikacji zorientowanych obiektowo oraz jest wewnętrznym językiem programowania. W porównaniu do innych języków np. C++, gdzie programista musi zaprojektować nawet najmniejszy fragment, język Visual Basic umożliwia korzystanie z gotowych elementów systemu - interfejsu wejścia-wyjścia, gotowych funkcji obsługi zdarzeń. W tym języku przygotowane zostały dwie aplikacje projektu. Pierwszą z nich jest menu wyboru aplikacji do zainstalowania, które inicjuje się po starcie płyty w napędzie, drugą jest przeglądarka dodatkowych zdjęć znajdujących się na płycie, która instalowana jest wraz z głównym programem "SKŁADACZ".
W obydwu przypadkach, podczas tworzenia aplikacji, przypisano wcześniej stworzone ikony odpowiednio - kursorom myszy i poszczególnym aplikacjom. Umieszczono również przygotowane przyciski oraz tła aplikacji. Przedstawiona poniżej funkcja jest integralną częścią kodu obydwu aplikacji i służy ona detekcji litery napędu, w którym znajduje się płyta z aplikacją.

```
Public Function Test() As String

Dim chaslo As String
Dim Dysk As String
Dim i As Integer
Dim szukane(25) As String
Dim jest As Boolean

'Lista przeglądanych dysków

szukane(1) = "C:"
szukane(2) = "D:"
szukane(3) = "E:"
szukane(4) = "F:"
szukane(5) = "G:"
szukane(6) = "H:"
szukane(7) = "I:"
szukane(8) = "J:"
szukane(9) = "K:"
szukane(10) = "L:"
szukane(11) = "M:"
szukane(12) = "N:"
szukane(13) = "O:"
szukane(14) = "P:"
szukane(15) = "Q:"
szukane(16) = "R:"
szukane(17) = "S:"
szukane(18) = "T:"
szukane(19) = "U:"
szukane(20) = "V:"
szukane(21) = "W:"
szukane(22) = "X:"
szukane(23) = "V:"
szukane(24) = "Z:"

i = 1

Do While i $<=$ 24

'Funkcja porównywania nazwy dysku

If Dir(szukane(i), vbVolume) = "skl" Then
chaslo = szukane(i)

i = 30
jest = True

Test = chaslo

Else
i = i + 1
End If

Loop

If jest = False Then
```

```
    'Tutaj realizowana jest funkcja zależna
    ' od programu, w przypadku aplikacji instalacyjnej
    'Realizowana jest funkcja Uninstall,
    ' w przeglądarce realizowane jest okno dialogowe.

End If

End Function
```

Zarówno jedna jak i druga aplikacja opierają się głównie na podstawowych elementach i zdarzeniach zawartych w języku Visual Basic.

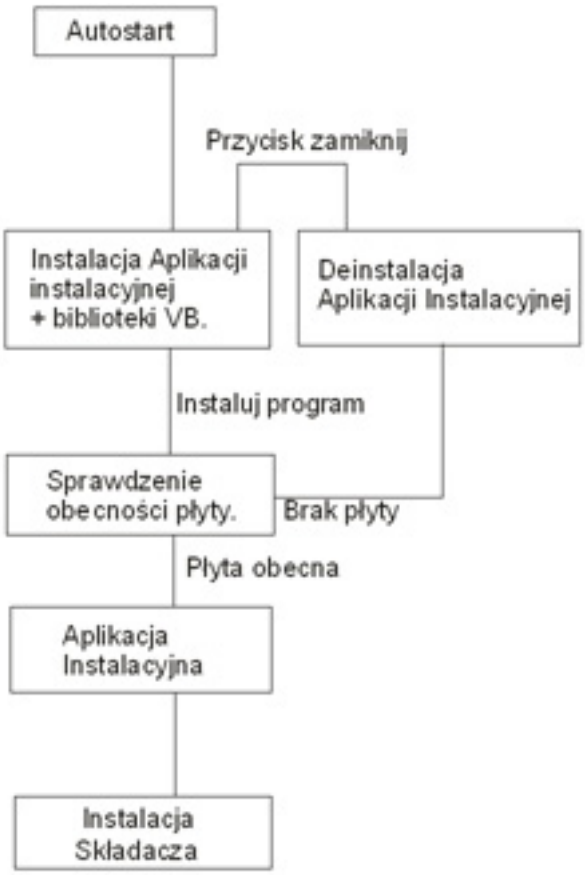
6.4 Programy instalacyjne

Program Inno Setup 3 jest aplikacją służącą do tworzenia aplikacji instalacyjnej, której tworzenie odbywa się za pomocą skryptów. Opis funkcji zawartych w skrypcie producent dostarcza wraz z programem. Oprócz tworzenia pliku skryptu, należy stworzyć także plik służący jako podstawa językowa dla programu - opis wszystkich menu, okien dialogowych. Ustalić również można formy, w jakich okna mają się pojawiać. Poniżej przedstawiamy fragment pliku skryptu oraz fragment pliku ustawień językowych.

```
; (Note: Scroll to the right to see the full lines!)
Source: "c:\vbfiles\stdole2.tlb";  DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile regtypelib
Source: "c:\vbfiles\msvbvm60.dll"; DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile regserver
Source: "c:\vbfiles\oleaut32.dll"; DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile regserver
Source: "c:\vbfiles\olepro32.dll"; DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile regserver
Source: "c:\vbfiles\asycfilt.dll"; DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile
Source: "c:\vbfiles\comcat.dll";   DestDir: "{sys}"; CopyMode:
     alwaysskipifsameorolder; Flags: restartreplace uninsneveruninstall
     sharedfile regserver
; end VB system files

[INI]
Filename: "{app}\Składacz.url"; Section: "InternetShortcut"; Key: "URL";
     String: "http:/www.pjwstk.edu.pl"

[Icons]
Name: "{group}\Składacz"; Filename: "{app}\Składacz.exe" ;WorkingDir: "{app}"
Name: "{group}\Składacz strona WWW"; Filename: "{app}\Składacz.url" ;
    WorkingDir: "{app}"
Name: "{group}\Przeglądarka"; Filename: "{app}\Przegladarka.exe" ;
    WorkingDir: "{app}"
Name: "{group}\Odinstaluj Składacz"; Filename: "{uninstallexe}" ;
    WorkingDir: "{app}"
Name: "{userdesktop}\Składacz"; Filename: "{app}\Składacz.exe"; Tasks:
```

```
    desktopicon ;WorkingDir: "{app}"

[Run]
Filename: "{app}\Składacz.exe"; Description: "Uruchom Składacza"; Flags:
     nowait postinstall skipifsilent

;============================================================================

; *** Misc. common
InformationTitle=Informacja
ConfirmTitle=Potwierdź
ErrorTitle=Błąd

; *** SetupLdr messages
SetupLdrStartupMessage=Instalacja programu: %1.%nCzy chcesz kontynuować ?
LdrCannotCreateTemp=Nie można stworzyć folderu tymczasowego.%nInstalacja anulowana.
LdrCannotExecTemp=Nie można uruchomić plików w folderze tymczasowym.

; *** Startup error messages
LastErrorMessage=%1.%n%nBłąd  %2: %3
SetupFileMissing=W folderze instalacyjnym brak pliku %1.%nProszę usunąć problem
    lub pobrać nową kopię programu.
SetupFileCorrupt=Plik instalacyjny jest uszkodzony.%nProszę pobrać nową kopię
    programu.
SetupFileCorruptOrWrongVer=Pliki instalacyjne są uszkodzone lub są niekompatybilne
    z tą wersją instalatora.%nProszę usunąć problem lub pobrać nową kopię instalatora.
NotOnThisPlatform=Tego programu nie można uruchomić w systemie  %1.
OnlyOnThisPlatform=Ten program można uruchomić wyłącznie w systemie %1.
WinVersionTooLowError=Program wymaga systemu  %1 w  wersji  %2  lub nowszej.
WinVersionTooHighError=Ten program nie może zostać zainstalowany w systemie  %1
  w wersji %2 oraz nowszej.
AdminPrivilegesRequired=Aby przeprowadzić instalację tego programu, Użytkownik musi
 być  zalogowany z uprawnieniami administratora.
SetupAppRunningError=Instalator wykrył, że program  %1  jest aktualnie uruchomiony.
    %n%nProszę zamknąć ten program i kliknąć 'OK', aby kontynuować lub 'Anuluj',
    aby przerwać instalację.
UninstallAppRunningError=Deinstalator wykrył, że program  %1  jest aktualnie
    uruchomiony.%n%nProszę zamknąć ten program i kliknąć 'OK', aby kontynuować
    lub 'Anuluj', aby przerwać deinstalację.

; *** Misc. errors
ErrorCreatingDir=Nie można utworzyć folderu "%1"
ErrorTooManyFilesInDir=Nie można zapisać pliku w folderze: "%1".%nFolder zawiera
     za dużo plików.
```

Aplikacja posiada również możliwość kompresowania plików, które będą instalowane. Podczas przygotowywania instalatora aplikacji "SKŁADACZ" okazało się, że konieczne jest stworzenie aplikacji umożliwiającej instalację programu Divix oraz Flash na docelowym komputerze. Po stworzeniu takiej aplikacji w języku Visual Basic niemożliwym było jej uruchomienie na komputerach nie posiadających bibliotek Visual Basic. Zatem koniecznością było stworzenie instalatora, który bez wiedzy użytkownika instaluje aplikację służącą do wyboru aplikacji do zainstalowania wraz z bibliotekami Visual Basic. Następnie, po wybraniu programu, automatycznie się ona odinstalowywuje. Za pomocą programu Inno Setup Creator stworzono dwie aplikacje instalacyjne. Schemat działania obydwu przedstawiony jest na rysunku 9.



Rysunek 9: Diagram stanów instalatora

6.5 Ikony i kursory

Ikony i kursory zostały wykonane w Axialis Iconworkshop 5.0. Program ten umożliwia zarządzanie oraz tworzenie ikon i kursorów pod wszystkie systemy operacyjne. Umożliwia też filtrowanie oraz importowanie plików z różnych programów graficznych.
W programie tym wykonano wszystkie ikony, kursory i przyciski wykorzystane w obydwu programach instalacyjnych. Programem, dzieki któremu możliwe było umieszczenie ikon w głównej części aplikacji "SKŁADACZ" był Icon Customizer. Umożliwia on zamianę ikon w dowolnym programie wykonywanym "exe".

6.6 Program *Region*

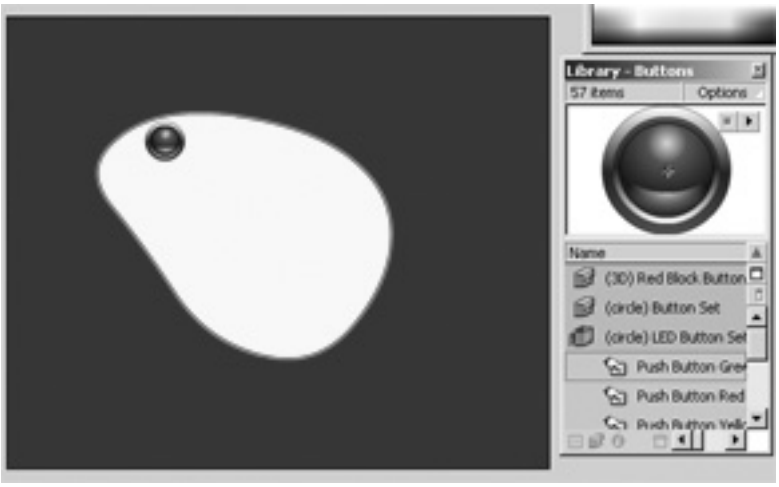
Program region służy do tworzenie plików .rgn, które następnie mogą być wykorzystane w głównej aplikacji do uzyskania innego niż prostokątny kształtu okna.

Możliwość tworzenia okien o zmiennych kształtach jest cechą wszystkich okien w systemie operacyjnym windows. Region jest zbiorem kwadratów (struktur typu RECT) określających obszar na którym aplikacja może być rysowana.

Program region umożliwia kilka sposobów utworzenia plików rgn. Zasadniczo jest on przystosowany do pracy z aplikacjami które mają mieć interfejs użytkownika stworzony we flashu.

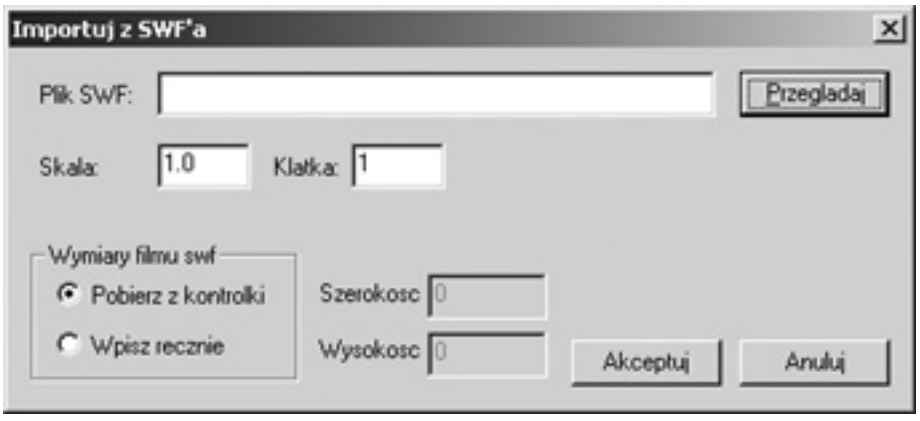
Poniżej znajduje się krótki opis kroków potrzebnych do stworzenia prostego interfejsu użytkownika o zmiennym kształcie.

1)Najpierw należy stworzyć w programie Flash, plik swf zawierający prosty kształt oraz przycisk który po naciśnięciu wysyła fscommende "exit".Patrz rysunek 6.



Rysunek 10: krok 1.Prosty kształt aplikacji, z przyciskiem zamykającym ją

2)Następnie otwieramy program Region i importujemy nowo powstały plik swf wybierając z menu plik - nowy - z swf'a. pokaże się okienko dialogowe Patrz rysunek 7.
W okienku tym możemy wybrać importowany plik swf. Określić skalę powiększenia go. Podać klatkę filmu, którą chcemy zaimportować. Można również określić wymiary filmu, bądź wybrać opcję "Pobierz z kontrolki" aby program sam je określił. Ta opcja przydaje się przy bardziej skomplikowanych interfejsach użytkownika, opcja automatyczna może czasami ustawić większą powierzchnię filmu swf niż jest ona w rzeczywistości. Po zaakceptowaniu ustawień program wygeneruje bitmape która pojawi się na ekranie w postaci nowego dokumentu.



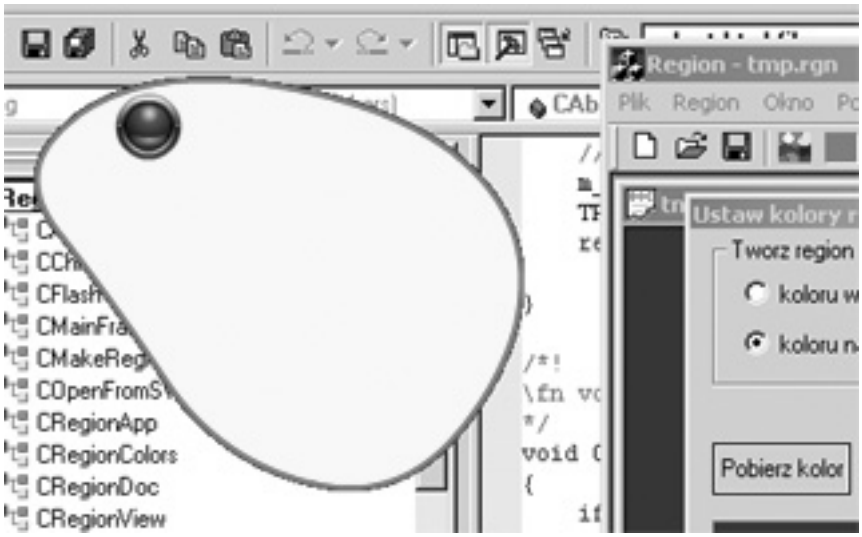
Rysunek 11: krok 2.Importowanie pliku swf

3)Teraz trzeba określić, które kolory stanowią wnętrze regionu. W tym celu należy otworzyć okienko narzędziowe wyboru kolorów : w menu region komenda kolory. Powinno ono być ustawione tak jak na rysunku 8.

Rysunek 12: krok 3.Okno wyboru kolorów regionu

4)Następnym etapem jest wybranie z menu plik, komendy zapisz, która otworzy okienko dialogowe, w którym należy podać nazwę tworzonego regionu.

5)Gdy region już się wygenerował można go przetestować: wybieramy z menu region komende testuj, następnie podajemy nazwę pliku zawierającego region. I okno testowe pokazuje się na ekranie, dodanie obszaru z akcją fscommand("Window Can Move"); dało by możliwość przemieszczania okna. Jeśli okienko testowe nie daje się zamknąć przyciskiem, można zamknąć je klawiszem enter.



Rysunek 13: krok 4.Test okna

Po zaimportowaniu pliku swf, można zapisać go do pliku bmp (menu region - zapisz do bmp). Który można edytować w dowolnym programie graficznym, a potem zaimportować do programu region, wybierając w menu plik - nowy - z bitmapy.

6.6.1 Strona techniczna programu

Program jest oparty na aplikacji typu MDI. Czyli umożliwia otwarcie wielu dokumentów na raz. Wybór MDI zamiast SDI (program może mieć otwarty tylko jeden dokument) ułatwi rozbudowę

aplikacji, jeśli w przyszłości będzie się chciało dodać interakcję między poszczególnymi regionami, jak np. kopiowanie czy porównywanie ich.

Do uzyskania bitmapy wybranej klatki z filmu swf, program Region korzysta z klasy CFlashDlg (używanej także w głównej aplikacji). Renderowanie do bitmapy polega na narysowaniu okna (które jest obiektem typu CFlashDlg) w kontekście urządzenia umieszczonym w pamięci. Kontekst urządzenia jest to klasa należąca do biblioteki funkcji windows. Udostępnia ona różne funkcje rysowania Z tych funkcji korzysta takze kontrolka shockwave która rysuje film flasha w oknie.

Plik region jest tworzony przy pomocy funkcji :

```
BOOL CMakeRegion::CreateFromBitmap(CDC \&memDC,float scale)
```

z klasy CMakeRegion. Funkcja ta jest uruchamiana w osobnym wątku, gdyż tworzenie regionu może być dość długo trwałe, co mogłoby zamrozić aplikacje.
Szczegółowy opis klas i funkcji znajdują się w postaci komentarzy w kodzie źródłowym.

7 Testy

Program był testowany w różnych wersjach systemu operacyjnego windows i nigdy nie sprawiał problemów. Systemami tymi były : Windows 98 (second edition), Windows 2000 Professional oraz Windows XP.

Trudności mogą wystąpić na słabszych komputerach (o mocy procesora mniejszej niż 450 MHz). Kontrolka ShockWaveFlash umożliwia ustawienie jakości renderowanego filmu. Można ją określić na małą, średnią oraz wysoką. Program stara się dobrać jakość renderowania filmu w zależności od mocy procesora. Odczytuje w tym celi ilość megaherców procesora z rejestru systemowego : HKEY_LOCAL_MACHINE/ Hardware/Description/System/CentralProcessor/0. W pliku config.ini są zdefiniowane dwie zmienne SWF_HIGH_QUALITY= i SWF_MEDIUM_QUALITY= które określają dla jakich procesorów jaka jest jakość renderowania plików swf.

8 Kompilacja projektu

Program "Składacz" był kompilowany pod Microsoft Visual C++ 6.0. Zawiera on większość wymaganych bibliotek i plików nagłówkowych. Dwie biblioteki trzeba jednak doinstalować. DirectX sdk i Platform sdk. Z pakietu DirectX sdk wykorzystywany jest tylko Direct Show. Natomiast Platform sdk zawiera pliki nagłówkowe potrzebne do kompilacji kodu obsługującego kontrolkę WebBrowser2 - pokazującą pliki html.

Jeśli pomimo instalacji wszystkich pakietów sdk , program nadal się nie kompiluje i pokazuje błąd związany z plikiem Transact.h, wówczas trzeba przekopiować plik Transact.h z katalogu Platform SDK/include do katalogu include Microsoft Visual C++.

9 Instalacja

Płytka po włożeniu do napędu, powinna sama uruchomić instalator. Umożliwia on doinstalowanie wymaganych elementów systemu, takich jak Flash Player oraz kodeki DivX 5.2. Jeśli instalacja nie zacznie się samoczynnie należy uruchomić instalator ręcznie z głównego katalogu płytki. Instalacja jest jest bardzo prosta. Użytkownik jest pytany o to, gdzie zainstalować aplikacje oraz czy utworzyć skrót na pulpicie.

Program umożliwia także deinstalację aplikacji.

10 Dokumentacja użytkowa

Po uruchomieniu program pokazuje ekran powitalny. Stąd możemy wybrać jeden z działów : obudowa , procesor , pamięć , płyta główna , napęd. Po wybraniu działu na ekranie pokażą się przyciski umożliwiające czytanie tekstu albo oglądanie filmu.

Na przykład po wybraniu działu o obudowach i naciśnięciu klawisza "film o obudowie" otworzy się okienko w którym będzie pokazany film (okienko to znajduje się na rysunku numer 2). Użytkownik może zmienić jego rozmiar poprzez kliknięcie i przytrzymanie prawego dolnego rogu okna, następnie w wyniku poruszania myszką okno będzie zmieniać wymiary. Film można zatrzymywać i przewijać. Okienko zamyka się przyciskając przycisk znajdujący sie w prawym górnym rogu. W czasie oglądania filmu można czytać tekst.

Kliknięcie przycisku znajdującego się w dolnym prawym rogu aplikacji powoduje powrót do strony powitalnej. Dwa przyciski znajdujące się w górnym prawym rogu to przycisk minimalizacji aplikacji oraz zamykania. Gdy aktywny jest tryb czytania tekstu, wówczas górny prawy przycisk służy do przejścia do poprzedniego ekranu aplikacji.

Literatura

[1] Antoni Diller, „LaTeXwiersz po wierszu ”,Helion ,Gliwice 2001. Z j. angielskiego przełożył Jan Jełowicki.

[2] Andrzej Jaszkiewicz, „Inżynieria oprogramowania”,Helion ,Gliwice 1997

[3] Bill Sergio, „Advanced UI”, `http://codeguru.earthweb.com/advancedui/Vskins.html`

[4] Jon Bates, Tim Tompkins, „Poznaj Visual C++ 6”,

[5] Peter Norton, „W sercu PC”,

[6] Charles Petzold, „Programowanie Windows”,

[7] Davis Chapman, „Visual C++ dla każdego”,

[8] Pomoc kontekstowa zawarta w programie Macromedia Flash,

[9] Bob Reselman,Richard Peasley,Wayne Pruchniak, „Poznaj Visual Basic 6”,

Skorowidz