

GEM — Plan testów

wersja 1.1

Maria Donten
Marek Grabowski
Piotr Hofman
Kuba Pochrybniak

Spis treści

1. Wprowadzenie	3
1.1. Cel	3
1.2. Zakres	3
1.3. Definicje i skróty	3
1.4. Załączniki	3
1.5. Omówienie reszty dokumentu	3
1.6. Uwagi ogólne	3
2. Testy ogólnych własności programu	3
2.1. Test uruchamiania i konfigurowania programu	3
3. Testy edytora graficznego	5
3.1. Testy podstawowych funkcjonalności	5
3.2. Test odporności na dużą liczbę obiektów	6
3.3. Test skomplikowanych zależności między obiektami	7
3.4. Test funkcjonalności przy dużym zagęszczeniu obiektów	7
3.5. Test podobieństwa między rysunkiem w edytorze i w przeglądarce	8
4. Testy edytora tekstowego	9
4.1. Test edycji tekstu	9
4.2. Testy zmiany kroju czcionki	11
5. Testy konwersji formatów	11
5.1. Test tłumaczenia obiektowego formatu danych na kod METAPOST	11
5.2. Test kompilacji kodu METAPOST	13
5.3. Test konwersji formatów	14
6. Testy przeglądarki efektów	14
6.1. Test odporności na skomplikowane rysunki	14
6.2. Test jakości wyświetlanych rysunków	15
6.3. Test wygody skalowania i przesuwania obrazka	16
6.4. Test podobieństwa między rysunkiem w edytorze i w przeglądarce	16
6.5. Test operowania na obrazkach o źródle innym niż GEM	16
7. Testy aplikacji sieciowej	17
7.1. Testy nawiązywania i zrywania połączeń	17
7.2. Test zmiany kierunku przepływu danych	18
7.3. Test zdalnego wywoływania odpowiednich metod	20
7.4. Testy reakcji na awarie sieci	21
8. Dane testowe	22
9. Historia zmian	22

1. Wprowadzenie

1.1. Cel

Celem tego dokumentu jest rozplanowanie przebiegu procesu testowania programu GEM oraz sprecyzowanie warunków określających, kiedy można uznać, że projekt spełnia wszystkie stawiane mu wymagania.

1.2. Zakres

Projekt GEM jest wykonywany w ramach zajęć Zespołowego Projektu Programistycznego na wydziale MIM UW. Celem procesu testowania programu GEM jest zweryfikowanie jakości programu, czyli pracy licencjackiej z informatyki — program i jego dokumentacja muszą spełniać wymogi stawiane pracy licencjackiej.

1.3. Definicje i skróty

Podane w załączniku „Słownik projektu GEM”, uaktualnianym na bieżąco, aby zawierał definicje wymagane przez powstające kolejno dokumenty.

1.4. Załączniki

— Dokument „Słownik projektu GEM”, wersja 4.1.

1.5. Omówienie reszty dokumentu

Pozostała część dokumentu jest opisem testów poszczególnych części programu. Rozdział drugi opisuje testy ogólnych własności programu, rozdział trzeci i czwarty odpowiednio testy edytora graficznego i tekstowego. Rozdział piąty opisuje planowane testy modułu odpowiedzialnego za konwersje między różnymi formatami wektorowymi. Rozdział szósty zawiera opis testów przeglądarki efektów, a rozdział siódmy testy sieciowej części aplikacji. Rozdział ósmy opisuje dane testowe używane przy testowaniu programu.

1.6. Uwagi ogólne

Ze względu na to, że część testowanych opcji dotyczy obsługi wieloelementowych rysunków i przekształcania ich na wiele różnych sposobów, często trudno dokładnie oceniać poprawność otrzymanych danych. Przy wielu testach Zespół zdecydował się więc badać poprawność działania programu dokładnie na małych danych, natomiast z pominięciem szczegółów dla dużych danych testowych (np. przy liczących kilkadziesiąt elementów rysunkach nie prowadzi się szczegółowego porównania obrazu w oknie edytora graficznego i w przeglądarce — osoba testująca ma tylko ocenić, czy nie widać istotnych rozbieżności).

2. Testy ogólnych własności programu

2.1. Test uruchamiania i konfigurowania programu

2.1.1. Opis testów

Aby można było wykonywać dalsze testy, trzeba najpierw upewnić się, czy można uruchamiać program bezbłędnie i czy jest dostępny interfejs do wszystkich funkcji programu.

Opisane zostały testy konfiguracji jedynie kilku podstawowych opcji. Drobiazgowe testowanie poprawności zmiany konfiguracji wszystkich parametrów poszczególnych części programu zajęłoby bardzo dużo czasu, a nie jest to kluczowe dla działania programu.

2.1.2. Przypadki użycia

- Zmiana konfiguracji programu,
- wyłączenie programu.

2.1.3. Założenia

Brak wstępnych założeń.

2.1.4. Przebieg

Test należy powtórzyć dla różnych systemów operacyjnych — wybranych wersji Windows i Linuksa. W każdym z systemów należy sprawdzić kilka zestawów danych konfiguracyjnych (przynajmniej 3).

Czynność	Oczekiwany wynik
start programu z systemu operacyjnego	pojawia się okno programu z uruchomionym edytorem graficznym; okno przeglądarki jest zwinięte (do prawej strony okna programu)
przejrzenie poszczególnych menu programu	udostępniane opcje zgodne z opisem zawartym w „Podręczniku użytkownika”
wyłączenie i ponowne włączenie paska narzędzi za pomocą opcji „Pasek narzędzi” w menu „Widok”	pasek narzędzi znika, a potem pojawia się w tym samym miejscu
wyłączenie i ponowne włączenie pola z komunikatami zewnętrznych narzędzi za pomocą opcji „Komunikaty” w menu „Widok”	pole tekstowe na dole okna programu pojawia się, a potem znika
rozwiniecie okna przeglądarki (poprzez użycie odpowiedniego suwaka), tak aby zajmowało połowę okna programu	okno przeglądarki zajmuje prawą połowę okna programu, pole graficzne zmienia kształt, aby zająć tylko lewą połowę
wybór z menu „Plik” opcji „Konfiguracja”	pojawia się okno edycji konfiguracji
wybór zakładki „Opcje ogólne”	widać wybraną zakładkę, z polami do podania wartości
zmiana następujących opcji: rozmiaru okna programu na „300 × 300”, rozmiaru okna przeglądarki na „25%” (szerokości ekranu), edytora pojawiającego się przy uruchomieniu programu na „tekstowy”; potwierdzenie zmiany opcji przyciskiem „Zastosuj”	okno programu zmienia rozmiar, przeglądarka zajmuje około 25% szerokości ekranu, nadal jest uruchomiony edytor tekstowy
wyłączenie programu	program jest wyłączony (należy sprawdzić, czy nie zostały pliki tymczasowe w katalogu roboczym programu)

ponowne uruchomienie programu	pojawia się okno programu — wymiary mają takie, jak tuż przed wyłączeniem; teraz włączony jest edytor tekstowy
ponowna zmiana konfiguracji i sprawdzenie efektów (powrót do punktu z włączeniem okna konfiguracji programu), lub wyłączenie programu	w zależności od wybranej czynności: zmiany wyglądu okna programu zgodnie z nowymi danymi konfiguracyjnymi lub wyłączony program

3. Testy edytora graficznego

3.1. Testy podstawowych funkcjonalności

3.1.1. Opis testów

W tej części testów będą badane standardowe funkcjonalności programu: otwieranie/zapis do pliku, wybór narzędzi, oglądanie rezultatów w przeglądarce etc.

3.1.2. Przypadki użycia

- Wczytanie i zapis edytowanego pliku,
- rysowanie podstawowych kształtów,
- szybki podgląd.

3.1.3. Założenia

Brak wstępnych założeń.

3.1.4. Przebieg

Czynność	Oczekiwany wynik
otwarcie nowego pliku	otwiera się okienko z nowym, pustym rysunkiem
narysowanie kilku podstawowych kształtów (tak, by łatwo się dały zapamiętać)	pojawienie się na ekranie narysowanych obiektów
zamknięcie edytowanego rysunku, bez wcześniejszego zapisu	pojawienie się okienka „Zapisz jako”
zapisanie pliku	pojawienie się informacji o zapisaniu pliku; zamknięcie okienka z rysunkiem
otwarcie zamkniętego przed chwilą rysunku	pojawienie się na ekranie dokładnie tego rysunku, który był zapisany
wybór i użycie w losowej kolejności wszystkich narzędzi do rysowania	pojawianie się na ekranie obiektów odpowiadających aktywnemu w danej chwili narzędziu
seria szybkich zmian na rysunku, przerywanych wybieraniem podglądu	szybkie odświeżanie rysunku w oknie przeglądarki

3.2. Test odporności na dużą liczbę obiektów

3.2.1. Opis testu

Program będzie działać na komputerach o różnorodnych parametrach i na różnych systemach operacyjnych. Na szybkich komputerach nie spodziewamy się żadnych problemów z wydajnością; na komputerach wolnych, wyposażonych w małą ilość pamięci, można spodziewać się problemów zarówno z szybkością działania, jak i zapychaniem pamięci oraz znaczącym spowalnianiem systemu operacyjnego.

Program powinien zostać przetestowany na następujących platformach (podane częstotliwości taktowania są orientacyjne):

- Celeron 600 MHz,
- Celeron 2 GHz,
- Pentium 600 MHz,
- Pentium 2 GHz,
- maszyna wirtualna na komputerze z procesorem o częstotliwości taktowania $\geq 1,5$ GHz,

zaś systemy, na których należy przetestować, to:

- Windows 98,
- Windows 2000,
- Windows XP,
- Linux (Debian),
- Linux (Fedora/Aurox),
- Linux (Gentoo).

Z powodu trudności technicznych program będzie testowany tylko na niektórych kombinacjach.

3.2.2. Przypadki użycia

- Rysowanie podstawowych kształtów,
- zaznaczanie zbioru obiektów.

3.2.3. Założenia

Przetestowane podstawowe funkcjonalności edytora graficznego.

3.2.4. Przebieg

Czynność	Oczekiwany wynik
tworzenie w dużym tempie nowych obiektów graficznych jednego rodzaju	pojawianie się obiektów na ekranie; w przypadku spowolnień pojawianie się z opóźnieniami
tworzenie w dużym tempie nowych obiektów graficznych przy częstych zmianach narzędzia	j.w.
zaznaczanie dużych grup obiektów; manipulacje na nich (przesuwanie, skalowanie etc.)	zadowalające tempo odświeżania obrazu
zamykanie z zapisem i ponowne otwieranie pliku z wieloma obiektami graficznymi	odpowiednie komunikaty; zapis i odczyt w sensownym czasie

3.3. Test skomplikowanych zależności między obiektami

3.3.1. Opis testów

Ponieważ program umożliwia pozycjonowanie względne obiektów względem innych obiektów, w dodatku bez ograniczenia na głębokość tego, trzeba przetestować, jak program się zachowuje przy tworzeniu skomplikowanych drzew takich właśnie zależności.

3.3.2. Przypadki użycia

- Rysowanie podstawowych kształtów,
- zaznaczanie zbioru obiektów,
- podstawowe przekształcenia na obiektach,
- kopiowanie, wklejanie oraz usuwanie obiektów.

3.3.3. Założenia

Przetestowane podstawowe funkcjonalności edytora graficznego.

3.3.4. Przebieg

Czynność	Oczekiwany wynik
tworzenie długiej serii obiektów o liniowej zależności położenia	pojawianie się właściwych w spodziewanych miejscach na ekranie
tworzenie serii obiektów o skomplikowanych zależnościach położenia; osiągnęte drzewo zależności ma mieć dosyć dużą szerokość oraz gałęzie bardzo różnej głębokości	brak niespodziewanych efektów
podstawowe manipulacje (przesuwanie, skalowanie etc.)	brak niespodziewanych efektów
usunięcie obiektu-korzenia drzewa	komunikat o istnieniu obiektów zależnych od usuwanego; prośba o potwierdzenie usunięcia; po usunięciu brak jakichkolwiek zmian pozycji na ekranie obiektów zależnych
usunięcie obiektu ze środka drzewa zależności	j.w.
usunięcie obiektu-liścia drzewa; po nim usunięcie jego ojca w drzewie	brak jakichkolwiek komunikatów

3.4. Test funkcjonalności przy dużym zagęszczeniu obiektów

3.4.1. Opis testów

Program nie ma ograniczenia na liczbę obiektów, które mogą pojawić się na rysunku. Prócz wydajności, problemem przy dużej ich liczbie może być zupełna nieczytelność rysunku. Program powinien umożliwiać wygodne działanie nawet w takich warunkach — dzięki łatwo dostępnej zmianie skali i przesuwaniu obrazka. Prócz tego ważna jest łatwo dostępna opcja zmiany wzajemnego położenia (przesłaniania) obiektów.

3.4.2. Przypadki użycia

- Rysowanie podstawowych kształtów,
- zaznaczanie zbioru obiektów,
- kopiowanie, wklejanie oraz usuwanie obiektów.

3.4.3. Założenia

Pozytywne wyniki przy testach wydajnościowych.

3.4.4. Przebieg

Czynność	Oczekiwany wynik
tworzenie dużej liczby obiektów, masowo nakładających się na siebie	brak niespodziewanych efektów; sensowna czytelność rysunku
zmiana skali (w szczególności przybliżanie), przesuwanie obrazka, zarówno suwakami, jak i „łapą”	szybkie odświeżanie ekranu
zaznaczenie losowego obiektu, przesuwanie go w górę i w dół oraz na wierzch i na spód w hierarchii kolejności wyświetlania	szybkie odświeżanie ekranu
zaznaczanie z góry upatrzzonego obiektu; w razie konieczności przybliżanie rysunku	wyświetlanie zaznaczenia właściwego obiektu

3.5. Test podobieństwa między rysunkiem w edytorze i w przeglądarce

3.5.1. Opis testów

Choć z założenia rysunek w edytorze graficznym ma być szkicowy, powinien jak najlepiej odzwierciedlać efekt, który zostanie osiągnięty poprzez późniejsze skompilowanie go w systemie METAPOST.

3.5.2. Przypadki użycia

- Szybki podgląd,
- pozycjonowanie podglądu względem rysunku w edytorze i vice versa.

3.5.3. Założenia

Stworzony rysunek składający się z podstawowych obiektów graficznych; wśród obiektów powinno znajdować się co najmniej kilka różnego rodzaju krzywych. Obiekty powinny mieć różne kolory i wypełnienia.

3.5.4. Przebieg

Czynność	Oczekiwany wynik
wybranie opcji pokazania rysunku w przeglądarce	pojawiający się po krótkim czasie obrazek w przeglądarce
wybranie opcji powiązania przeglądarki z edytorem; przesuwanie i skalowanie	szybkie odświeżanie obrazu w przeglądarce
wybranie opcji powiązania edytora z przeglądarką	szybkie odświeżanie obrazu w edytorze

[przy wszystkich powyższych testach]	efekt w przeglądarce wyraźnie podobny do szkicu w edytorze; w szczególności wszelkiego rodzaju krzywe powinny mieć identyczny przebieg
--------------------------------------	--

4. Testy edytora tekstowego

Jako że edytor tekstowy jest jednym z pierwszych części powstającego projektu będzie testowany w jego pierwszych fazach. Jako, że jest częścią w dużej mierze niezależną od reszty projektu, w późniejszych etapach będzie tylko i wyłącznie testowane wywoływanie różnych opcji udostępnianych przez edytor tekstowy a wymagających innych modułów.

Wymagane funkcjonalności edytora tekstowego to:

- Edycja tekstu,
- kompilowanie systemem METAPOST,
- zmiana używanego kroju czcionki.

Dalsza część rozdziału będzie kolejno omawiać kolejne grupy testów.

4.1. Test edycji tekstu

Ta część testów będzie się koncentrować na wszelkich problemach mogących pojawić się przy próbach edycji tekstu — podczas wczytywania, zapisu pliku, wklejania, cofania zmian. . .

Problemy dzielimy na:

- otwieranie, zamykanie i tworzenie nowych plików;
- pisanie tekstu;
- zaznaczanie, kasowanie, wycinanie, wstawianie, kopiowanie;
- cofanie i anulowanie cofania.

Testy będą odbywać się w środowisku nie przeciążonym działaniem innych programów i będą polegały na kolejnym wykonywaniu prostych czynności potwierdzających działanie edytora.

4.1.1. Opis testu

Test ma za zadanie sprawdzić możliwość edytowania tekstu.

4.1.2. Przypadki użycia

- Edycja tekstu.

4.1.3. Przebieg

Wszystkie czynności należy później powtórzyć dla skrótów klawiszowych oraz dla przycisków dostępnych na pasku narzędzi.

Czynność	Oczekiwany wynik
wejście w tryb edytora tekstowego	program informuje o wejściu w tryb tekstowy wyświetla się okno edytora tekstowego
wybranie w menu „Plik” — „Nowy”	pokazuje się nowy arkusz papieru
wpisanie w oknie tekstowym „ola ma kota”	w oknie powinno pojawić się „ola ma kota”

wybranie w menu „Plik” — „Zapisz jako...”	powinno otworzyć się okno pozwalające na podanie informacji o miejscu zapisania
podanie informacji o miejscu zapisania i wybranie opcji „Zapisz”	stworzenie pliku we wskazanym miejscu o tej nazwie i treści „ola ma kota”
dopisanie w oknie tekstowym „i psa” a następnie wybranie opcji zapisu	dodanie do stworzonego pliku tekstowego „i psa”
zaznaczenie fragmentu tekstu i wybranie w menu „Edycja” — „Kopiuj”	zaznaczony fragment powinien znaleźć się w schowku
wybranie w menu „Plik” — „Nowy”	powinna powstać zakładka z nowym arkuszem i ona powinna być otwarta
ustawienie kursora w polu tekstowym i wybranie w menu „Edycja” — „Wstaw”	tekst ze schowka pojawia się w oknie tekstowym
w menu „Plik” wybór „Zapisz”	powinno się pojawić okno wyboru pliku do zapisu
zapisanie pliku w nowym miejscu a następnie wybranie w menu „Plik” — „Otwórz”	powinno otworzyć się menu pozwalające na wybór pliku do otworzenia
wybór pliku tekstowego	pojawia się nowy arkusz zawierający otwierany plik
w menu „Edycja” wybieranie opcji zaznaczenia wszystkiego	cały tekst powinien zostać podświetlony
w menu „Edycja” wybieranie opcji wycięcia	tekst zostaje usunięty z arkusza i zapisany do schowka
w menu „Plik” wybieranie opcji „Zamknij”	zakładka zostaje zamknięta
przejsięcie do wcześniejszej zakładki i wybranie opcji „Wklej” z menu „Edycja”	cały tekst ze schowka powinien zostać wklejony
zaznaczenie fragmentu tekstu i wybieranie opcji „Usuń” z menu „Edycja”	kawałek tekstu zostaje usunięty ale schowek się nie zmienia
wybranie w menu „Edycja” — „Wklej”	zostaje wklejony kawałek tekstu znajdujący się w schowku
zaznaczenie kawałka tekstu i w menu „Edycja” wybieranie opcji „Kopiuj” a następnie ustawienie kursora i wybieranie opcji „Wklej”	zaznaczony tekst zostaje dodany we wskazanym miejscu
w menu „Edycja” wybieranie „Cofnij”	wklejony tekst znika
w menu „Edycja” wybór opcji „Ponów”	wklejony tekst powinien zostać ponownie wklejony
dopisanie „ala ma też słonia i hipcia” a następnie 5 razy operacje Cofnij	kolejno dopisywane wyrazy powinny zniknąć po każdym kliknięciu
w menu „Edycja” kliknięcie 5 razy „Ponów”	kolejno dopisywane wyrazy powinny pojawiać się po każdym kliknięciu
wybieranie opcji „Zamknij” z menu „Plik”	program kończy swoje działanie testy zakończyły się pomyślnie

4.2. Testy zmiany kroju czcionki

Ta część testów będzie się koncentrować na testach polegających na zmianie ustawień wyglądu czcionki.

Testy będą odbywać się w środowisku nie przeciążonym działaniem innych programów, i będą polegały na kolejnym wykonywaniu prostych czynności potwierdzających działanie edytora.

4.2.1. Opis testu

Test ma za zadanie sprawdzić możliwość dostosowania wyglądu tekstu do potrzeb autora.

4.2.2. Przypadki użycia

— Zmiana kroju czcionki.

4.2.3. Założenia

Edytor tekstowy przeszedł podstawowe testy.

4.2.4. Przebieg

Wszystkie czynności należy później powtórzyć dla skrótów klawiszowych oraz dla przycisków dostępnych na pasku narzędzi. Test należy powtórzyć dla kilku czcionek i wielkości, zwracając uwagę na polskie litery.

Czynność	Oczekiwany wynik
wybranie w menu „Narzędzia” opcji „Preferencje”	wyświetla się okno konfiguracji
wybranie w oknie ustawienia czcionki, a następnie zmiana wielkości i kroju czcionki i zatwierdzenie	następuje zmiana wyglądu dokumentu

Testy należy powtórzyć 10 razy, dla różnych kombinacji ustawianych opcji.

5. Testy konwersji formatów

Kolejne trzy scenariusze testów tworzą serię — połączenie ich istotnie ułatwia pracę, pozwala wykonywać wczytywanie i zapis plików mniejszą liczbą razy i umożliwia lepsze sprawdzenie poprawności wyników działania każdego z testowanych modułów.

5.1. Test tłumaczenia obiektowego formatu danych na kod METAPOST

5.1.1. Opis testów

Poniżej opisane są planowane testy działania modułu tłumaczącego wewnętrzny format danych programu na kod METAPOST. Najbardziej istotna jest poprawność tłumaczenia, należy sprawdzić także zachowanie modułu dla dużych danych wejściowych. Zespół nie planuje natomiast (przynajmniej narazie) dokonywać tłumaczenia w taki sposób, aby otrzymany kod METAPOST był elegancki i optymalny.

Testowanie modułu tłumaczącego wymaga użycia kompilatora języka METAPOST niezwiązanego z programem GEM.

5.1.2. Przypadki użycia

— Tłumaczenie wewnętrznego formatu danych na kod METAPOST.

5.1.3. Założenia

Odbyły się (i zakończyły sukcesem) testy edytora graficznego, istnieją pliki graficzne w wewnętrznym formacie danych programu GEM.

5.1.4. Przebieg

Wszystkie czynności należy wykonać dla skrótów klawiszowych oraz dla przycisków dostępnych na pasku narzędzi. Test powtarzamy 20 razy, na niektórych danych testowych kilkakrotnie, aby sprawdzić, jakie mogą być różnice efektów pomiędzy wykonaniami. Należy przetestować kilka bardzo dużych plików.

Czynność	Oczekiwany wynik
otwarcie pliku graficznego w wewnętrznym formacie danych	otwiera się okno z rysunkiem wczytanym z pliku
wybranie opcji tłumaczenia wewnętrznego formatu danych na kod METAPOST	jeśli okno z komunikatami było niewidoczne, pojawia się na ekranie; zostaje wypisany komunikat o rozpoczęciu tłumaczenia; po chwili wyświetla się również komunikat o zakończeniu tłumaczenia, z podaniem ścieżki pliku wynikowego; nie powinny wystąpić błędy
kompilacja pliku wynikowego za pomocą kompilatora języka METAPOST (niekończnie związane z programem GEM)	kompilacja powinna zakończyć się pomyślnie — wygenerowany automatycznie kod musi być poprawny
otworzenie otrzymanego po kompilacji rysunku za pomocą dowolnego programu obsługującego format eps w celu porównania efektu z początkowym rysunkiem w formacie obiektowym (można użyć przeglądarki programu GEM, jeśli została przetestowana)	rysunki muszą się zgadzać co do liczby, kształtu i innych własności obiektów, a także ich położenia względem siebie, z odpowiednią dokładnością

5.1.5. Uwagi

Po zakończeniu tego scenariusza należy przejść do następnego i wykonać go na uzyskanym wynikowym pliku z kodem METAPOST, jeśli po drodze nie wystąpił żaden błąd.

Należy również przetestować opcję tłumaczenia na pustym rysunku — powinien zostać wyświetlony komunikat o błędzie.

5.2. Test kompilacji kodu METAPOST

5.2.1. Opis testu

Ta część testów dotyczy wywołania kompilatora METAPOST. Test ma za zadanie sprawdzić połączenie zewnętrznego narzędzia — kompilatora kodu METAPOST — z programem GEM.

5.2.2. Przypadki użycia

— Kompilowanie systemem METAPOST.

5.2.3. Założenia

Testy edytora tekstowego zakończyły się pozytywnie.

5.2.4. Przebieg

Wszystkie czynności należy wykonać dla skrótów klawiszowych oraz dla przycisków dostępnych na pasku narzędzi. Test powtarzamy 20 razy, na niektórych danych testowych kilkakrotnie, aby sprawdzić, jakie mogą być różnice efektów pomiędzy wykonaniami. Należy przetestować kilka bardzo dużych plików.

Czynność	Oczekiwany wynik
otworzenie pliku tekstowego z kodem METAPOST	otwiera się zakładka z wczytanym tekstem
wybór opcji kompilacji kodu METAPOST	jeśli okno z komunikatami było niewidoczne, pojawia się na ekranie; zostaje wypisany komunikat o rozpoczęciu kompilacji; jeśli nie wystąpią błędy, po chwili wyświetla się również komunikat o zakończeniu kompilacji, z podaniem ścieżki pliku wynikowego; jeśli kod nie jest poprawny, pojawia się komunikat o błędzie i przerwaniu kompilacji
jeśli nie było błędów kompilacji: otworzenie pliku wynikowego za pomocą dowolnego programu graficznego obsługującego format eps	wyświetla się odpowiadający kodowi METAPOST rysunek

5.2.5. Uwagi

Jeśli nieoczekiwanie wystąpiły błędy kompilacji, wynikające z niepoprawności kodu, raczej nie powinno to świadczyć o nieprawidłowym podłączeniu kompilatora do programu GEM. Wyjątkiem są sytuacje, kiedy błąd wynika z braku pliku źródłowego lub niemożności utworzenia pliku wynikowego. Ponadto, jeśli błędem zakończy się test kompilacji przeprowadzany jest na kodzie wygenerowanym automatycznie przez moduł tłumaczący programu GEM, należy uwzględnić możliwość nieprawidłowego działania tego modułu.

5.3. Test konwersji formatów

5.3.1. Opis testów

W tej części testów sprawdzane będą opcje konwersji rysunku (wyjściowo w formacie eps) na inne formaty wektorowe. Testy należy wykonać kilkakrotnie (przynajmniej 3 razy) dla każdego formatu wektorowego, do którego konwersję udostępnia program GEM.

5.3.2. Przypadki użycia

— Kompilowanie systemem METAPOST.

5.3.3. Założenia

Testy edytora tekstowego zakończyły się pozytywnie.

5.3.4. Przebieg

Czynność	Oczekiwany wynik
wybór opcji „Konwersja formatów” z menu „Narzędzia”	otwiera się okienko konwersji formatów
wybór formatu docelowego z udostępnionej przez program listy; wpisanie w pole tekstowe ścieżki do pliku źródłowego oraz ścieżki do pliku wynikowego; potwierdzenie akcji	rozpoczyna się proces konwersji; po jego zakończeniu pojawi się informacja o sukcesie

5.3.5. Uwagi

Jeśli proces konwersji zostanie przerwany błędem, nie należy tego od razu traktować jako błąd implementacji programu GEM. Należy zbadać, czy błąd wynika z nieprawidłowego przekazania ścieżek plików przez główny program do konwertera, czy powstał podczas samego procesu tłumaczenia pliku. Pierwszy przypadek oznacza wykrycie błędu programu GEM, drugi — konieczność sprawdzenia działania modułu konwersji na innych danych testowych. Dopiero w przypadku pojawiania się błędów przy konwersji wielu różnych danych testowych trzeba uznać, że konwerter został nieprawidłowo podłączony do programu lub źle funkcjonuje w danym środowisku.

6. Testy przeglądarki efektów

6.1. Test odporności na skomplikowane rysunki

6.1.1. Opis testów

Przeglądarka ma być podręcznym narzędziem, z możliwością odświeżania co chwila podczas pracy z edytorem. Dlatego ważne jest, by działała szybko, również przy dużych i skomplikowanych rysunkach.

6.1.2. Przypadki użycia

- Szybki podgląd,
- powiększanie i zmniejszanie obrazka,
- przesuwanie obrazka.

6.1.3. Założenia

Stworzony rysunek składający się z dużej liczby obiektów graficznych.

6.1.4. Przebieg

Przebieg testów ma być analogiczny jak w punkcie 3.5, ze szczególnym zwróceniem uwagi na wydajność.

6.2. Test jakości wyświetlanych rysunków

6.2.1. Opis testów

Ważne jest, by wyświetlany w przeglądarce rysunek w miarę możliwości w stu procentach odzwierciedlał efekt, który pojawi się przy wstawianiu rysunku do konkretnej pracy naukowej. Dlatego do testu użyty będzie nie tylko sam program, ale również kompilator L^AT_EX-a.

6.2.2. Przypadki użycia

- Szybki podgląd,
- powiększanie i zmniejszanie obrazka,
- przesuwanie obrazka.

6.2.3. Założenia

Stworzony rysunek składający się z podstawowych obiektów graficznych; rysunek w miarę różnorodny, z dużą liczbą etykiet. Wielkość rysunku powinna być taka, by zmieścił się w odpowiednim miejscu pracy matematycznej.

6.2.4. Przebieg

Czynność	Oczekiwany wynik
skompilowanie rysunku i wyświetlenie go w przeglądarce	brak niespodziewanych efektów
włączenie rysunku do pracy matematycznej w formacie pdf (skompilowanie PDF _{L^AT_EX} -em); porównanie efektów w przeglądarce programu GEM (po odpowiednim przeskalowaniu) oraz w programie Adobe Reader	identyczne bądź prawie identyczne efekty
włączenie rysunku do pracy matematycznej w formacie ps (skompilowanie L ^A T _E X-em); porównanie efektów w przeglądarce programu GEM (po odpowiednim przeskalowaniu) oraz w programie GhostView	identyczne rozmiary i kształty rysunku; ze względu na brak wygładzania w programie GhostView obrazek w przeglądarce GEM powinien być istotnie wyższej jakości
wydruk pracy (z obrazkiem stworzonym i skompilowanym przez GEM) w pdf; porównanie wydruku z widokiem w przeglądarce	z dokładnością do kolorów wydruk powinien być niemal identyczny

6.3. Test wygody skalowania i przesuwania obrazka

6.3.1. Opis testów

Przeglądarka, choć z założenia ma służyć do oglądania efektów w całości (w większości przypadków rysunki naukowe mają się mieścić na maksimum 1/3 kartki formatu A4, często z dużymi marginesami) ma umożliwiać jak najwygodniejsze oglądanie szczegółów obrazka.

6.3.2. Przypadki użycia

- Szybki podgląd,
- powiększanie i zmniejszanie obrazka,
- przesuwanie obrazka.

6.3.3. Założenia

Stworzony duży rysunek składający się z podstawowych obiektów graficznych; wśród obiektów powinno znajdować się co najmniej kilka różnego rodzaju krzywych. Obiekty powinny mieć różne kolory i wypełnienia.

6.3.4. Przebieg

Czynność	Oczekiwany wynik
seria zmian skali w obie strony	szybkie odświeżanie obrazu
duże powiększenie obrazka; seria przesunięć suwakami	szybkie odświeżanie obrazu
duże powiększenie obrazka; seria przesunięć „łapą”	szybkie odświeżanie obrazu

6.4. Test podobieństwa między rysunkiem w edytorze i w przeglądarce

6.4.1. Opis testów

Test został opisany w punkcie 3.5.

6.5. Test operowania na obrazkach o źródle innym niż GEM

6.5.1. Opis testów

Przeglądarka ma być dodatkiem do edytora graficznego, ale powinna też działać na „zewnętrznych” plikach.

6.5.2. Przypadki użycia

- Szybki podgląd,
- powiększanie i zmniejszanie obrazka,
- przesuwanie obrazka.

6.5.3. Założenia

Rysunek w kodzie METAPOST wzięty „z zewnątrz”, już istniejący, stworzony za pomocą narzędzi niezależnych od programu GEM.

6.5.4. Przebieg

Czynność	Oczekiwany wynik
otwarcie pliku	pojawienie się obrazka na ekranie
testy jak w punkcie 6.3	efekty jak w punkcie 6.3

7. Testy aplikacji sieciowej

Testowanie aplikacji sieciowej będzie się odbywać po testach standardowych funkcjonalności programu, więc zakładamy że wszystkie funkcje lokalne działają poprawnie. W związku z tym, w trakcie testowania aplikacji sieciowej, nie będzie sprawdzane poprawne wykonywanie zdalnie wywołanych poleceń, a jedynie sam fakt poprawnego ich wywołania.

Wymagane funkcjonalności sieciowej części aplikacji to:

- nawiązywanie i zrywanie połączenia,
- zmiana kierunku przepływu danych,
- zdalne wywoływanie odpowiednich metod,
- reakcje na awarie sieci,

Dalsza część rozdziału będzie kolejno omawiać kolejne grupy testów.

7.1. Testy nawiązywania i zrywania połączeń

Ta część testów będzie się koncentrować na wszelkich problemach mogących pojawić się przy próbach nawiązania połączenia.

Problemy dzielimy na:

- błędy aplikacji,
- niedostępność sieci (urządzenia sieciowego),
- działający firewall, blokujący połączenie,
- zajętość portu,
- znaczne opóźnienie w sieci.

Test podstawowy składać się będzie z:

1. Nawiązania połączenia na domyślnym porcie.
2. Rozłączenia poprawnego.
3. Nawiązania połączenia na dowolnym porcie.
4. Niepoprawnego zerwania połączenia.

Testy będą polegać na próbach nawiązania połączenia — na początku w „zdrowym” środowisku — w celu wykrycia błędów aplikacji. W chwili kiedy aplikacja będzie działała poprawnie, do środowiska zaczną być wprowadzane różne czynniki zakłócające. Po wprowadzeniu każdej modyfikacji do środowiska, następować będzie powtórzenie standardowego zestawu testów. W razie wykrycia jakichś problemów będą wykonywane dodatkowe testy, mające na celu zlokalizowanie błędu.

7.1.1. Opis testu

Test ma za zadanie sprawdzić poprawność nawiązywania i kończenia połączenia przez dwie komunikujące się aplikacje, w różnych warunkach środowiskowych.

7.1.2. Przypadki użycia

- Połączenie w celu oglądania rysunku,
- połączenie w celu rysowania,
- rozłączenie.

7.1.3. Założenia

Wykonano wszystkie lokalne (niesieciowe) testy programu.

7.1.4. Przebieg

Czynność	Oczekiwany wynik
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
zakończenie połączenia ze strony serwera	pojawia się informacja o zakończeniu połączenia
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
zakończenie połączenia ze strony klienta	pojawia się informacja o zakończeniu połączenia
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
twarde, nieprawidłowe rozłączenie ze strony serwera	pojawia się informacja o utraceniu łączności
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
twarde, nieprawidłowe rozłączenie ze strony klienta	pojawia się informacja o utraceniu łączności

Po skończeniu testów na standardowym porcie zostaje uruchomiona ta sama seria testów na innym. Po zakończeniu serii testów wprowadzamy zmiany w środowisku w następującej kolejności:

1. Włączamy firewalla (w szczególności „zaporę systemu Windows”).
2. Uruchamiamy jakiś inny program zajmujący dany port.
3. Uruchamiamy dużo innych programów generujących znaczny ruch w sieci.
4. Odłączamy kabel sieciowy.

Po wprowadzeniu każdej zmiany przeprowadzamy całą serię testów (w wypadku zmian nr 1 i 3), albo sprawdzamy jak program reaguje na niemożliwość wykonania polecenia (w wypadku zmian nr 2 i 4).

7.2. Test zmiany kierunku przepływu danych

Ta klasa testów będzie rozwinięciem testów nawiązywania i zrywania połączenia. Różnica między tymi funkcjonalnościami polega na braku wpływu użytkownika na ustalanie

adresu (i/lub portu). W związku z tym testy będą wyglądać praktycznie identycznie jak testy nawiązywania połączenia, przy czym podstawowy zestaw testów wygląda następująco:

1. Nawiązanie połączenia na standardowym porcie.
2. Zmiana kierunku przepływu danych (kilkukrotna).
3. Poprawne zakończenie połączenia.
4. Nawiązanie połączenia na dowolnym porcie.
5. Zmiana kierunku przepływu danych (kilkukrotna).
6. Niepoprawne zerwanie połączenia.

Tak samo jak w poprzednim przypadku, w razie napotkania problemów, będą przeprowadzane dodatkowe testy mające na celu zlokalizowanie błędu w programie.

7.2.1. Opis testu

Test ma za zadanie sprawdzić poprawność zachowania aplikacji sieciowej po zmianie kierunku przepływu danych

7.2.2. Przypadki użycia

- Połączenie w celu oglądania rysunku,
- połączenie w celu rysowania,
- zmiana kierunku przepływu danych,
- rozłączenie.

7.2.3. Założenia

Wykonano wszystkie lokalne (niesieciowe) testy programu oraz testy 7.1.

7.2.4. Przebieg

Czynność	Oczekiwany wynik
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
Zmian kierunku przepływu danych ze strony klienta	pojawia się informacja o zamianie klienta z serwerem
zakończenie połączenia ze strony serwera	pojawia się informacja o zakończeniu połączenia
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
Zmian kierunku przepływu danych ze strony klienta	pojawia się informacja o zamianie klienta z serwerem
zakończenie połączenie ze strony klienta	pojawia się informacja o zakończeniu połączenia
rozpoczęcie nasłuchiwania na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080

podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
Zmian kierunku przepływu danych ze strony serwera	pojawia się informacja o zamianie klienta z serwerem
zakończenie połączenia ze strony serwera	pojawia się informacja o zakończeniu połączenia
rozpoczęcie nasłuchiwanie na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
Zmian kierunku przepływu danych ze strony serwera	pojawia się informacja o zamianie klienta z serwerem
zakończenie połączenia ze strony klienta	pojawia się informacja o zakończeniu połączenia

7.3. Test zdalnego wywoływania odpowiednich metod

Ten zestaw testów wymaga poprawnie działającego połączenia. Polegać będzie na nawiązaniu połączenia (i ewentualnej zmianie kierunku przepływu danych), a następnie wywołaniu wszystkich metod rysowania równolegle ze sprawdzaniem, czy zdalnie wywołane metody odpowiadają tym wywołanym lokalnie, na komputerze nadającym.

Metody wymagające przetestowania to głównie funkcjonalności edytora graficznego, czyli między innymi:

- rysowanie prymitywów,
- obsługa stosu wycofań,
- obsługa grup obiektów.

7.3.1. Opis testu

Test ma za zadanie sprawdzić poprawność zachowania aplikacji sieciowej przy zdalnym rysowaniu.

7.3.2. Przypadki użycia

- Połączenie w celu oglądania rysunku,
- połączenie w celu rysowania,
- zdalne wykonywanie instrukcji,
- rozłączenie.

7.3.3. Założenia

Wykonano wszystkie lokalne (niesieciowe) testy programu oraz testy 7.1.

7.3.4. Przebieg

Czynność	Oczekiwany wynik
rozpoczęcie nasłuchiwanie na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080

podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
próba narysowania czegoś na serwerze	brak efektów - rysowanie powinno być zablokowane
powtórzenie testów edytora graficznego ze strony klienta	efekty takie same jak przewidziane w testach - zarówno po stronie klienta jak i serwera
zakończenie połączenia ze strony serwera	pojawia się informacja o zakończeniu połączenia

7.4. Testy reakcji na awarie sieci

W tej części procesu testowania będą sprawdzane reakcje systemu na awarie sieci. W praktyce będzie to nagłe zrywanie połączenia — np. odłączanie kabla sieciowego — gdyż protokół TCP/IP ma zapewniać bezbłędne przesyłanie pakietów. Testy będą polegały y na powtórzeniu poprzedniego etapu testowania (zdalnego wywoływania metod) z jednoczesnym rozłączaniem sieci w losowych momentach.

W razie wykrycia błędu w konkretnej metodzie będą aplikowane kolejne testy mające na celu zlokalizowanie konkretnego błędu w kodzie programu.

7.4.1. Opis testu

Test ma za zadanie sprawdzić poprawność zachowania aplikacji sieciowej przy zdalnym rysowaniu.

7.4.2. Przypadki użycia

- Połączenie w celu oglądania rysunku,
- połączenie w celu rysowania,
- zdalne wykonywanie instrukcji,
- rozłączenie.

7.4.3. Założenia

Wykonano testy 7.1.

7.4.4. Przebieg

Czynność	Oczekiwany wynik
rozpoczęcie nasłuchiwanie na standardowym porcie (8080)	zostaje uruchomiony serwer nasłuchujący na porcie 8080
podłączenie klienta do serwera na standardowym porcie (8080)	pojawia się informacja o nawiązaniu połączenia
rozpoczęcie zdalnego rysowania ze strony klienta	systematyczne pojawianie się wyników pracy klienta na serwerze
twarde zerwanie połączenia (softwarowe lub hardwarowe)	pojawienie się komunikatu o zerwaniu połączenia po upływie 20 sekund

Test powtarzany 20 razy, przy użyciu różnych narzędzi do rysowania.

8. Dane testowe

Testowanie programu GEM wymaga danych testowych dwóch rodzajów:

1. Rysunków zapisanych za pomocą języka METAPOST do testów edytora tekstowego i kompilacji.
2. Rysunków w plikach graficznych lub na wydrukach, do testów edytora graficznego i konwersji obiektowego formatu danych do kodu METAPOST.

Pliki z kodem języka METAPOST są konieczne, aby sprawdzić działanie edytora tekstowego — w szczególności dobrać domyślne parametry wyświetlania tekstu tak, aby był on czytelny, oraz ewentualnie przetestować podświetlanie składni języka METAPOST (funkcjonalność opcjonalna). Inne właściwości edytora tekstowego mogą zostać sprawdzone na dowolnym pliku tekstowym.

Ponadto gotowego kodu METAPOST wymagają testy kompilatora. Nie będzie sprawdzana sama poprawność kompilacji, ponieważ program GEM będzie korzystał z zewnętrznego narzędzia — Zespół nie tworzy własnego kompilatora. Należy przetestować tylko poprawność podłączenia kompilatora do programu, w szczególności wczytywanie pliku po kompilacji przez przeglądarkę i wyświetlanie komunikatów o błędach kompilacji.

Najprawdopodobniej uda się dostać gotowe pliki z rysunkami wykonanymi w języku METAPOST od znanych członkom Zespołu użytkowników języka METAPOST. Jeśli nie, Zespół sam stworzy pliki do testów.

Rysunki do testów edytora graficznego powinny zostać zaczerpnięte z prac matematycznych i informatycznych, aby jak najlepiej sprawdzić zgodność działania programu z wyznaczonymi przez Zespół celami. Podczas testów edytora graficznego Zespół zamierza korzystać z własnej bazy przykładowych ilustracji prac matematycznych, powstałej podczas badań nad zapotrzebowaniami klientów i tworzenia wizji projektu. Osoby testujące będą tworzyć kopię danego rysunku za pomocą programu GEM i porównywać otrzymany obraz z wzorem. Zespół oczekuje, że dokładność powstałych ilustracji nie będzie gorsza od oryginału, a istotnej poprawie w stosunku do wzoru ulegnie wygląd oznaczeń tekstowych na rysunkach.

Powstałe przy testowaniu działania edytora graficznego rysunki, zapisane w wewnętrznym formacie danych programu, zostaną wykorzystane do testowania tłumaczenia obiektowego formatu danych na kod METAPOST, a następnie, po przetłumaczeniu, będzie można wykorzystać te pliki jako część materiałów do testów edytora tekstowego i kompilatora.

9. Historia zmian

Wersja	Data	Autorzy zmian	Zmiany
0.5	14.03.2007	cały Zespół	pierwotna, niepełna wersja
0.8	15.03.2007	Marysia Donten	pierwsze poprawki całości tekstu
1.0	17.03.2007	Kuba Pochrybniak	ostateczne poprawki
1.1	31.03.2007	Kuba Pochrybniak	poprawki językowe