

GEM — Plan zarządzania projektem

wersja 1.2

Maria Donten
Marek Grabowski
Piotr Hofman
Kuba Pochrybniak

Spis treści

1. Wprowadzenie	4
1.1. Cel	4
1.2. Zakres	4
1.3. Definicje i skróty	4
1.4. Załączniki	4
1.5. Omówienie reszty dokumentu	4
2. Omówienie projektu	4
2.1. Cel i zakres projektu	4
2.1.1. Cel	4
2.1.2. Zakres dostarczonych funkcjonalności	4
2.2. Założenia i zależności	5
2.2.1. Założenia dotyczące produktu	5
2.2.2. Założenia i zależności dotyczące pracy zespołu	5
2.3. Produkty projektu	5
2.4. Procedury zmian w planie projektu	6
2.4.1. Ulepszanie rozwiązań w projekcie	6
2.4.2. Opóźnienia	6
2.4.3. Nieprzewidziane trudności	7
3. Organizacja projektu	7
3.1. Struktura organizacyjna	7
3.2. Kontakty zewnętrzne	7
3.3. Role i zadania	8
4. Zarządzanie projektem	10
4.1. Oszacowania	10
4.2. Plan projektu	10
4.2.1. Plan faz projektu	10
4.2.2. Cele poszczególnych iteracji	11
4.2.3. Wydania	11
4.2.4. Harmonogram projektu	12
4.2.5. Zasoby	13
4.2.6. Budżet	13
4.3. Plany iteracji	13
4.4. Nadzór i kontrola projektu	14
4.4.1. Plan zarządzania harmonogramem	14
4.4.2. Plany dotyczące zapewniania i kontroli jakości oraz oceny produktu	15
4.4.3. Plan raportów	15
4.5. Plan zarządzania ryzykiem	15
4.6. Plan zamknięcia projektu	15
5. Plany procesów technicznych	16
5.1. Programowanie	16
5.2. Metody, narzędzia i stosowane technologie	16
5.3. Plan akceptacji produktu	16
6. Plany pomocnicze	17
6.1. Plan zarządzania zmianami	17
6.2. Plan dokumentacji	17
6.2.1. Dokumentacja podstawowa	17
6.2.2. Dokumentacja dodatkowa	17

7. Dodatek A — uaktualniony harmonogram prac	18
7.1. Uwagi ogólne	18
7.2. Podział pracy	18
8. Historia zmian	20

1. Wprowadzenie

1.1. Cel

Dokument ten ma na celu przedstawienie planu prac nad projektem GEM oraz informacji i założeń dotyczących projektu, uzasadniających przyjęty plan.

1.2. Zakres

Niniejszy dokument dotyczy projektu mającego na celu stworzenie programu wspomagającego tworzenie prostej grafiki do prac naukowych (szczególnie z zakresu nauk ścisłych). Program ma zapewniać wygodny interfejs pozwalający na łatwe tworzenie podstawowych typów rysunków pojawiających się w różnych pracach. Produkt będzie oferował możliwość tworzenia grafiki wektorowej i konwersji jej do języka METAPOST — dobrze współpracującego z systemem L^AT_EX — najpopularniejszym narzędziem do pisania prac naukowych.

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”.

1.5. Omówienie reszty dokumentu

W kolejnych rozdziałach tego dokumentu przedstawiono:

- krótki opis projektu,
- kwestie organizacyjne, w tym konkretne metody pracy nad projektem,
- plan realizacji projektu, w tym wstępny harmonogram prac.

2. Omówienie projektu

2.1. Cel i zakres projektu

2.1.1. Cel

Celem projektu GEM jest dostarczenie osobom piszącym prace matematyczne lub informatyczne wygodnego narzędzia do tworzenia standardowej dla prac naukowych grafiki. Większość pracowników naukowych nie tworzy rysunków do prac na tyle często, żeby warto z ich perspektywy było poświęcać dużo czasu i pracy na poznanie jakiegoś specjalistycznego języka do tworzenia grafiki (np. METAPOST czy Asymptote). Jednak korzystanie z popularnych programów graficznych nie daje zadowalających efektów, chociażby dlatego, że te programy nie są przystosowane do współpracy z L^AT_EX-em i dodawania tekstu matematycznego do rysunków.

2.1.2. Zakres dostarczonych funkcjonalności

W pierwszym wydaniu:

1. Graficzny edytor do rysowania prostych rysunków geometrycznych, z silnym wsparciem rysowania grafów (podstawowe funkcjonalności wymienione w Wizji).

2. Przeglądarka efektów umożliwiająca wygodny podgląd wykonywanego obrazka.
3. Edytor tekstowy języka **METAPOST** pozwalający zarówno na edycję rysunku od początku w trybie tekstowym, jak i na wprowadzanie zmian do kodu **METAPOST** generowanego przez program na podstawie rysunku tworzonego z użyciem graficznego interfejsu. Edytor jest zintegrowany z resztą systemu, a przede wszystkim połączony z przeglądarką efektów.
4. Narzędzie do przesyłania bieżących zmian na rysunku przez sieć.
5. Konwerter do graficznych formatów wektorowych: pdf, eps, jpg, png.
[Nie trzeba przypadkiem podzielić tego na dwa wydania, czy cus?]

2.2. Założenia i zależności

2.2.1. Założenia dotyczące produktu

1. Produkt jest przewidziany do użytkowania na komputerach osobistych.
2. Na komputerze użytkownika musi być zainstalowana maszyna wirtualna Javy zgodna z SUN JRE 1.5 oraz dystrybucja systemu $\text{T}_{\text{E}}\text{X}$ zawierająca $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ **METAPOST** i podstawowe konwertery.
3. Zalecany jest system operacyjny Windows 2000/XP lub nowszy.
4. Komputer, na którym używany jest program GEM nie wymaga podłączenia do sieci (oczywiście jeśli użytkownik nie chce korzystać z graficznej komunikacji sieciowej).

2.2.2. Założenia i zależności dotyczące pracy zespołu

Zespół Programistyczny „Drużyna A” składa się z czterech osób i ze względu na to, że projekt jest wykonywany w ramach zajęć Zespołowego Projektu Programistycznego, Zespół nie może się powiększyć.¹ Wobec tego zakres projektu musi być dostosowany do liczby osób i przeznaczonego na niego czasu, czyli bieżącego roku akademickiego. Plan zarządzania powstaje przy założeniu, że żaden z członków Zespołu nie zrezygnuje z udziału w projekcie w trakcie roku akademickiego. Punkt 4.5 określa, jakie działania zostaną podjęte, jeśli taka sytuacja jednak nastąpi.

Ponadto przyjmujemy następujące założenia co do pracy programistów:

1. Każdy z członków zespołu może przeznaczyć na projekt około 120 godzin w semestrze letnim, czyli w fazie kodowania i testów.
2. Każdy z programistów będzie pisał w środowisku Eclipse IDE.
3. Każdy z programistów będzie używał SVN.
4. Każdy z członków zespołu będzie przestrzegał procedur zmian w projekcie (opisanych w punkcie 2.4).

2.3. Produkty projektu

1. Dokumentacja projektu opisana w punkcie 6.2.
2. Edytor graficzny podstawowy służący do tworzenia nieskomplikowanej i nie wyspecjalizowanej grafiki, pozwalający na konwersję do **METAPOST** i dalszą do popularnych formatów graficznych. Wraz z przeglądarką i edytorem kodu języka **METAPOST**. Prócz prostych rysunków edytor będzie posiadał silne wsparcie do rysowania grafów.
3. Format danych służący do przechowywania rysunków w formie obiektowej (najprawdopodobniej XML).

¹ Być może po zakończeniu zajęć projekt będzie nadal rozwijany i wtedy skład Zespołu ulegnie istotnym zmianom, ale ten dokument nie planuje szczegółów rozwoju projektu po zakończeniu bieżącego roku akademickiego.

4. Narzędzie do przesyłania bieżących zmian na rysunku przez sieć.
5. Instrukcja obsługi (podręcznik użytkownika) wraz z kilkoma przykładami możliwości programu.

2.4. Procedury zmian w planie projektu

2.4.1. Ulepszanie rozwiązań w projekcie

Przyczyna Znalezienie nowego lepszego sposobu rozwiązującego pewne zagadnienie

Zmiana Zmiana sposobu realizacji zagadnienia

Procedura postępowania

1. Zasygnalizowanie problemu na forum Zespołu.
2. Przesłanie do kierowniczki projektu zarysu informacji o rozważanych zmianach.
3. Przesłanie do wszystkich osób, dla których ta zmiana może mieć znaczenie, informacji o pomysłe wprowadzenia zmiany.
4. Po konsultacjach na forum i zatwierdzeniu nowego sposobu realizacji przez kierowniczkę osoba zgłaszająca (lub inna wydelegowana przez kierowniczkę) wprowadza zmiany w dokumentacji. Należy zwrócić szczególną uwagę na wpływ proponowanych zmian na sposób realizacji pozostałych części projektu, nie tylko tej, w której następuje zmiana.
5. Nowa dokumentacja jest zatwierdzana przez kierowniczkę.
6. Zmiana wchodzi w życie, jeśli poprzednie punkty zostały spełnione. W przeciwnym przypadku aktualne pozostają niezmienione plany.

2.4.2. Opóźnienia

Przyczyna Opóźnienie w realizacji projektu

Zmiana Zmiany w planie rozwoju projektu

Procedura postępowania Po wykryciu opóźnienia należy rozważyć następujące aspekty:

1. Jak duże będzie opóźnienie, jeśli nie zostaną podjęte kroki zaradcze?
2. Czy w wyniku tego opóźnienia nastąpi blokada rozwoju projektu poprzez opóźnienie testów już gotowych fragmentów lub niemożliwość pisania innych funkcjonalności?
3. Czy oddelegowanie dodatkowej osoby do realizacji zadania spowoduje rozwiązanie problemu i czy można taką osobę wydelegować?
4. Czy założony w dokumentacji sposób rozwiązania części projektu będącej przyczyną opóźnienia jest słuszny i czy nie należy dokonać zmian w założeniach projektowych? Jeśli tak, to należy przejść do punktu 2.4.3.

Możliwe do podjęcia decyzje:

1. Dodatkowe zmotywowanie osoby odpowiedzialnej za opóźnienie, poprzez rozmowę (tylko w przypadku, gdy opóźnienie wynika z niedotrzymywania terminów przez członków Zespołu, a nie z błędu projektowego).
2. Oddelegowanie do realizacji dodatkowej osoby. Należy wówczas zmodyfikować harmonogram rozwoju, biorąc pod uwagę czas potrzebny na wdrożenie dodatkowej osoby w realizowany fragment. Można również podzielić niezrealizowane zadanie na podzadania i podzielić odpowiedzialność, ale nie jest to konieczne — współpracujące osoby mogą uzgodnić to między sobą.
3. Modyfikacja harmonogramu pracy i wykorzystanie czasu zapasowego, przeznaczonego na realizację części funkcjonalności opcjonalnych.
4. Rozmowa z prowadzącą zajęcia o możliwości rezygnacji z problematycznej części ze względu na bardzo poważne, nieprzewidziane trudności (podejmowana w ostateczności).

2.4.3. Nieprzewidziane trudności

Przyczyna Wyjście na jaw nieprzewidzianej wcześniej zależności lub trudności uniemożliwiającej realizację projektu zgodnie ze specyfikacją.

Zmiana Zmiana projektu

Procedura postępowania

1. Członek Zespołu, który wykrył błąd projektowy, powiadamia o tym resztę Zespołu, w szczególności kierowniczkę Zespołu.
2. Zespół ustala sposoby rozwiązania problemu: spisuje wszystkie pomysły, wybiera dwa lub trzy najbardziej sensowne i spisuje dokładnie ich zalety i wady.
3. Kierownik po konsultacji z członkami Zespołu wybiera jeden ze sposobów do dokładnego opracowania.
4. Kierownik Zespołu zgłasza problem prowadzącej zajęcia lub oddelegowuje do tego inną osobę. Pokazuje prowadzącej opracowanie proponowanego rozwiązania.
5. Jeśli propozycje zmian zostaną zatwierdzone przez prowadzącą, to następuje zmiana w dokumentacji. Zmiany wprowadza osoba wydelegowana przez kierownika. Wszyscy członkowie grupy zostają powiadomieni o zakończeniu procedury zmiany. Zmiany zostają wdrożone do realizacji.
6. Jeśli propozycje zmian nie zostaną zatwierdzone, Zespół ustala z prowadzącą zajęcia inne rozwiązanie problemu, satysfakcjonujące obie strony.

Uwaga *Można konsultować się z prowadzącą również na pośrednich stadiach naprawy błędu projektowego.*

3. Organizacja projektu

3.1. Struktura organizacyjna

W skład Zespołu programistycznego Drużyna A wchodzi następujące osoby:

Maria Donten — kierowniczka Zespołu

Marek Grabowski

Piotr Hofman

Jakub Pochrybniak

Zadania poszczególnych członków Zespołu oraz części projektu, za które są odpowiedzialni, zostały wyszczególnione w punkcie 3.3.

3.2. Kontakty zewnętrzne

Program GEM jest przeznaczony przede wszystkim dla studentów oraz pracowników naukowych zajmujących się matematyką lub informatyką. Członkowie Zespołu są studentami matematyki i informatyki na wydziale MIM UW, wobec czego znają wiele osób, zarówno spośród studentów, także studiów doktoranckich, jak i wśród pracowników naukowych. Szczególnie cenne są znajomości wśród doktorantów, ponieważ:

- piszą oni istotnie więcej tekstów naukowych niż większość studentów studiów magisterskich, wobec czego częściej potrzebują narzędzia umożliwiającego łatwą edycję grafiki przeznaczonej do tego typu prac;
- dosyć często dają korepetycje, więc mogą ocenić przydatność i jakość aplikacji sieciowej programu GEM (umożliwiającej przesyłanie przez sieć dokonywanych na rysunku zmian w czasie rzeczywistym);

— ze względu na bardziej bezpośrednie kontakty niż ze starszymi pracownikami naukowymi, dużo łatwiej będzie znaleźć osoby chętne do testowania programu pośród doktorantów. Ze względu na to, że społeczność wydziału MIM UW nie jest grupą zorganizowaną, nie można określić osób odpowiedzialnych za kontakty z Zespołem ze strony społeczności wydziału. Osobą z Zespołu odpowiedzialną za kontakty ze studentami i pracownikami wydziału MIM UW jest Piotr Hofman.

Warto w tym miejscu zaznaczyć, że pomysł programu GEM powstał na bazie zapotrzebowania członków Zespołu na narzędzie do tworzenia grafiki matematycznej. Na podstawie dotychczasowych doświadczeń możemy dobrze określić, jakie funkcjonalności są najbardziej potrzebne i jak powinien być zaprojektowany interfejs, żeby program był wygodny. Co oczywiście nie umniejsza znaczenia kontaktów zewnętrznych, szczególnie jeśli chodzi o testowanie programu oraz funkcjonalności dodatkowe, bardziej specjalistyczne.

3.3. Role i zadania

Zestawienie pod względem osób

Wszyscy członkowie Zespołu	<ul style="list-style-type: none"> • Biorą udział w spotkaniach i dyskusjach dotyczących projektu; • aktywnie uczestniczą w pisaniu dokumentacji projektu (a także czytają części dokumentów napisane przez inne osoby, w celu zminimalizowania liczby błędów); • są programistami, tworzą kod programu; • biorą udział w testowaniu programu.
Maria Donten	<ul style="list-style-type: none"> • Jako kierowniczką Zespołu jest odpowiedzialna za całość projektu; • odpowiada za podział pracy nad dokumentacją; • dba o spójność i poprawność merytoryczną dokumentacji, aczkolwiek ma w tym udział cały Zespół, czytając dokładnie dokumenty; • kieruje pracami nad edytorem tekstowym języka METAPOST i przeglądarką efektów; (1) • uczestniczy w rozwoju edytora, zajmuje się zmianami rozmiarów i przesuwaniem obrazka, własnościami etykiet, podstawową wersją siatek; (2) • kieruje pracami nad konwersją wewnętrznego formatu danych do kodu METAPOST; (3) • programuje dodawanie punktów do krzywych, przekształcenia afiniczne i inne elementy grafiki przeznaczone do utworzenia w czwartej iteracji; (4) • bierze udział w pracach nad aplikacją sieciową; (4) • jest odpowiedzialna za wygląd interfejsu graficznego programu. (4+)

Marek Grabowski	<ul style="list-style-type: none"> • Ma za zadanie zapoznawać pozostałych członków Zespołu z technologiami używanymi w projekcie, które nie wszyscy znają (w miarę potrzeb); • jest odpowiedzialny za kampanię reklamową programu GEM; • implementuje pierwotną (szkicową) wersję edytora grafiki; (1) • uczestniczy w rozwoju edytora, dba o interfejs, pracuje nad własnościami etykiet; (2) • kieruje pracami nad konwersją rysunku utworzonego przez METAPOST do innych formatów wektorowych; (3) • kieruje pracami nad aplikacją sieciową. (4)
Piotr Hofman	<ul style="list-style-type: none"> • Jest odpowiedzialny za kontakty ze studentami i pracownikami wydziału MIM — w razie potrzeby uzyskania informacji od potencjalnych użytkowników GEM kieruje zdobywaniem tych informacji; • w końcowej fazie projektu zajmie się znalezieniem kilku osób chętnych do testowania programu; • bierze udział w programowaniu edytora tekstowego języka METAPOST i przeglądarki efektów; (1) • uczestniczy w rozwoju edytora, zajmuje się implementacją menedżera obiektów, kontroli wzajemnego przesłaniania obiektów i zarządzania kolorami; (2) • kieruje pracami nad rozwojem modułów edytora do rysowania grafów i diagramów; (3) • kieruje pracami nad dodawaniem punktów do krzywych, przekształceniami afinicznymi i innymi elementami grafiki przeznaczonymi do utworzenia w czwartej iteracji; (4) • jest odpowiedzialny za podręcznik użytkownika. (4)
Kuba Pochrybniak	<ul style="list-style-type: none"> • Dbą o poprawność językową i skład tekstu dokumentacji; • ma za zadanie zapoznać resztę Zespołu z potrzebnymi technikami grafiki wektorowej oraz ze szczegółami systemu METAPOST; • kieruje pracami nad pierwotną (szkicową) wersją edytora grafiki; (1) • kieruje pracami nad rozwojem edytora, implementuje krzywe Béziera; (2) • uczestniczy w programowaniu konwersji wewnętrznego formatu danych do kodu METAPOST; (3) • uczestniczy w pracach nad dodawaniem punktów do krzywych, przekształceniami afinicznymi i innymi elementami grafiki przeznaczonymi do utworzenia w czwartej iteracji. (4)

**Zestawienie zadań programistycznych i zajmujących się nimi podzespołów
(w nawiasie numer iteracji)**

Zadanie	Osoba odpowiedzialna	Inni wykonawcy
Pierwotna (szkicowa) wersja edytora grafiki. (1)	Kuba Pochrybniak	Marek Grabowski
Edytor tekstowy języka METAPOST z przeglądarką efektów. (1)	Maria Donten	Piotr Hofman
Rozwój możliwości edytora grafiki — krzywe Béziera, wygodne zarządzanie obiektami i kolorami, siatki. (2)	Kuba Pochrybniak	Maria Donten, Marek Grabowski, Piotr Hofman
Konwersja wewnętrznego formatu danych do kodu METAPOST. (3)	Maria Donten	Kuba Pochrybniak
Konwersja rysunku utworzonego przez METAPOST do innych formatów wektorowych. (3)	Marek Grabowski	—
Dalszy rozwój własności edytora — moduły do rysowania grafów. (3)	Piotr Hofman	—
Aplikacja sieciowa. (4)	Marek Grabowski	Maria Donten
Zaawansowane własności edytora — dodawanie punktów do krzywych, przekształcenia afiniczne. (4)	Piotr Hofman	Maria Donten, Marek Grabowski, Kuba Pochrybniak

4. Zarządzanie projektem

4.1. Oszacowania

Projekt będzie wykonany przez studentów, w ramach uniwersyteckich zajęć „Zespołowy Projekt Programistyczny”, na własnych komputerach studentów.

Szacowany czas trwania prac to 8 miesięcy (3,5 miesiąca przeznaczone na fazę projektową, pozostałe 4,5 na tworzenie oprogramowania). Po tym okresie ma być gotowa podstawowa wersja projektu, być może z częścią funkcjonalności opcjonalnych. Dalsze funkcjonalności opcjonalne będą dodawane w okresie późniejszym, już nie w ramach zajęć uniwersyteckich.

Nowe oszacowania projektu mogą się pojawić w przypadku zmiany decyzji o opcjonalności poszczególnych funkcji, jak również na skutek konsultacji z potencjalnymi użytkownikami.

Zespół szacuje czas potrzebny do wykonania projektu na 60 osobodni. Stworzenie podstawowej funkcjonalności programu wymaga około 48 osobodni, zgodnie z oszacowaniami przedstawionymi w dodatku do wizji. Ponadto Zespół planuje przeznaczyć przynajmniej 12 osobodni na wykonywanie tych funkcjonalności opcjonalnych, które chcielibyśmy zamieścić już w wersji 1.0 programu, pod warunkiem, że nie powstaną żadne opóźnienia.

4.2. Plan projektu

4.2.1. Plan faz projektu

Pracę nad projektem można podzielić na następujące fazy:

nr	faza	kiedy	kryterium sukcesu
1	analiza i projektowanie	październik–styczeń	<ul style="list-style-type: none"> • została uzgodniona dokładna zawartość projektu • powstała wymagana dokumentacja
2	programowanie	luty–maj	<ul style="list-style-type: none"> • działający edytor ze wszystkimi podstawowymi funkcjonalnościami • działająca konwersja do systemu METAPOST • działająca przeglądarka
3	testowanie i zdanie oprogramowania	czerwiec	<ul style="list-style-type: none"> • testowanie przebiegło zgodnie z planem • program działa poprawnie • program spełnia oczekiwania pod względem funkcjonalności

4.2.2. Cele poszczególnych iteracji

Pierwsza iteracja Celem pierwszej iteracji jest zaimplementowanie edytora w bardzo podstawowej formie, z elementarnym interfejsem. Pierwsza iteracja zakończy się w lutym.

Druga iteracja Celem drugiej iteracji jest rozwinięcie edytora do formy, w której będzie spełniał wszystkie podstawowe funkcjonalności. Druga iteracja jest najdłuższa — zajmie cały marzec i połowę kwietnia.

Trzecia iteracja Trzecia iteracja jest poświęcona współpracy edytora z systemem METAPOST, czyli konwersji wewnętrznego formatu danych do kodu języka METAPOST, kompilowaniu oraz konwersji gotowych rysunków do różnych formatów graficznych. Będą również rozwijane funkcjonalności wspomagające rysowanie grafów. Trzecia iteracja będzie trwać od połowy kwietnia do połowy maja. Po jej zakończeniu zostanie wydana wersja β programu.

Czwarta iteracja Celem czwartej iteracji jest uruchomienie modułu sieciowego oraz dodanie części funkcjonalności opcjonalnych. Moduł sieciowy zostanie uruchomiony do końca maja — wtedy rozpocznie się faza testowania programu. Inne funkcjonalności opcjonalne będą powstawać równolegle z częścią testów, do połowy czerwca (będziemy intensywnie testować program bez funkcjonalności opcjonalnych). Następnie odbędzie się ostateczne testowanie wersji 1.0 programu.

Kolejne iteracje W przyszłości prawdopodobnie Drużyna A lub jej część zechce rozwijać projekt o kolejne funkcjonalności wymienione w dokumentacji jako opcjonalne, jak również wszelkie inne, które pojawią się wskutek konsultacji z potencjalnymi użytkownikami.

4.2.3. Wydania

Przewidywane jest wydanie wersji β w okolicach połowy maja 2007 i rozpowszechnienie jej wśród dobranej wcześniej grupy znanych Zespołowi osobiście matematyków i informatyków.

Wydanie pełnej wersji (1.0) produktu przewidywane jest na koniec czerwca 2007.

Daty kolejnych wydań będą zależeć od rozwoju projektu, a co za tym idzie, od zainteresowania nim środowisk naukowo-studenckich.

W harmonogramie uwzględnione zostały również te funkcjonalności opcjonalne, które Zespół planuje dodać do programu jak najszybciej. Zakładając, że w fazie programowania nie będzie żadnych opóźnień, chcielibyśmy już w maju i w czerwcu uzupełnić edytor o możliwości wykonywania przekształceń afinicznych na obiektach i dodawania punktów do istniejących krzywych (co umożliwi dodatkowe zmiany ich kształtów).

4.2.4. Harmonogram projektu

funkcjonalność	kiedy	iter.
Rysowanie podstawowych obiektów geometrycznych (proste, odcinki, łamane otwarte i zamknięte, prostokąty, okręgi)	luty	1
Możliwość edycji kodu rysunku w formacie METAPOST, wraz z przeglądarką efektów	luty	1
Możliwość przesuwania całego rysunku	marzec	2
Możliwość powiększania i pomniejszania wyświetlanego obrazu	marzec	2
Dołączanie etykiet do obrazka	marzec	2
Rysowanie krzywych Béziera za pomocą zaznaczania punktów, przez które krzywa powinna przechodzić	marzec	2
Przejrzysty, łatwy w obsłudze interfejs	marzec	2
Możliwość modyfikacji parametrów istniejących obiektów (np. położenie, kolor)	marzec	2
Możliwość określania względnego lub bezwzględnego położenia obiektów	marzec	2
Sterowanie kolejnością wyświetlania obiektów, a więc i ich wzajemnym przesłanianiem (połączone z menedżerem obiektów)	kwiecień	2
Możliwość wykorzystania siatki do pozycjonowania punktów i etykiet	kwiecień	2
Możliwość operowania na podzbiorach obiektów (przesuwania, zmiany koloru itp.)	kwiecień	2
Możliwość dodawania do rysunku już istniejących i zapamiętanych struktur	kwiecień	2
Możliwość sterowania wyglądem i kolorem krawędzi i wierzchołków	kwiecień	3
Możliwość sterowania przebiegiem krawędzi	kwiecień	3
Transformacja obiektowego formatu danych na kod METAPOST	maj	3
Transformacja formatu METAPOST do formatów pdf, eps, jpg, png	maj	3
Możliwość przeciągania wierzchołków, z automatycznym przeciąganiem krawędzi	maj	3
Możliwość wklejania tła dla rysunku	maj	4
Możliwość rysowania przez sieć	maj	4
Obsługa przekształceń afinicznych na obiekcie	czerwiec	4
Możliwość dodawania punktów do istniejących obiektów oraz ich zmiany za pomocą tych punktów	czerwiec	4
Możliwość znajdowania punktów przecięć krzywych	później	5+
Wzbogacenie edytora tekstowego o podświetlanie składni języka METAPOST	później	5+
Pakiet pozwalający na rysowanie wykresów funkcji jednej zmiennej	później	5+

Pakiet pozwalający na rysowanie wykresów funkcji dwóch zmiennych	później	5+
Możliwość wykonywania prostych konstrukcji geometrycznych	później	5+
Baza schematów rysunków	później	5+

4.2.5. Zasoby

Plan zatrudnienia Nad programem będzie pracowało czworo członków Drużyny A. Nie przewiduje się zatrudniania dodatkowych osób. Po zakończeniu zajęć ZPP prawdopodobnie część Zespołu będzie chciała w dalszym ciągu rozwijać projekt. Wówczas zostanie rozważone włączenie do projektu nowych programistów.

4.2.6. Budżet

Projekt powstaje jako darmowe oprogramowanie w ramach zajęć z Zespołowego Projektu Programistycznego. Nie jest przewidywany żaden budżet.

4.3. Plany iteracji

Każda z iteracji jest podzielona na mniej więcej dwutygodniowe etapy. Przed każdymi zajęciami ZPP w drugim semestrze zostanie stworzony raport podsumowujący ostatni etap. Należy zaznaczyć, że pierwszy etap pierwszej iteracji Zespół planuje zakończyć przed rozpoczęciem drugiego semestru, natomiast drugi etap czwartej, w większości opcjonalnej iteracji, zakończy się w trakcie sesji, po zakończeniu zajęć.

iter.	etap	co zostanie wykonane
1	1	<ul style="list-style-type: none"> Początkowa wersja podstawowego edytora grafiki (będzie jakaś interakcja z użytkownikiem, ale niekoniecznie możliwość rysowania obiektów geometrycznych). Edytor tekstowy języka METAPOST.
1	2	<ul style="list-style-type: none"> Edytor grafiki pozwalający na rysowanie prostych obiektów geometrycznych. Przeglądarka efektów połączona z edytorem. Koniec pierwszej iteracji, następują testy wewnątrz Zespołu.
2	1	<ul style="list-style-type: none"> Możliwość przesuwania całego rysunku. Możliwość powiększania i pomniejszania wyświetlanego obrazu. Pierwsza wersja dodawania etykiet do rysunku. Pierwsza wersja rysowania krzywych Béziera.
2	2	<ul style="list-style-type: none"> Zaawansowana wersja interfejsu graficznego programu (później może nastąpią niewielkie poprawki, jeśli będzie to potrzebne). Gotowe dodawanie etykiet do obrazka. Gotowa obsługa krzywych Béziera. Możliwość modyfikacji parametrów istniejących obiektów (np. położenie, kolor). Możliwość określania względnego lub bezwzględnego położenia obiektów. Pierwsza wersja obsługi siatek. Dokładny projekt menedżera obiektów, ewentualnie jego pierwsza wersja.

2	3	<ul style="list-style-type: none"> • Możliwość dodawania do rysunku już istniejących i zapamiętanych struktur. • Gotowe możliwości wykorzystania siatki do pozycjonowania punktów i etykiet. • Menedżer obiektów. • Sterowanie kolejnością wyświetlania obiektów, a więc i ich wzajemnym przesłanianiem się. • Możliwość operowania na podzbiorach obiektów (przesuwania, zmiany koloru itp.). • Koniec drugiej iteracji, program jest testowany przez członków Zespołu.
3	1	<ul style="list-style-type: none"> • Rozpoczęte prace nad transformacją obiektowego formatu danych na kod METAPOST. • Rozpoczęte prace nad transformacją formatu METAPOST do formatów pdf, eps, jpg, png. • Sterowanie wyglądem i kolorem krawędzi i wierzchołków. • Możliwość sterowania przebiegiem krawędzi.
3	2	<ul style="list-style-type: none"> • Gotowa transformacja obiektowego formatu danych na kod METAPOST. • Gotowa transformacja formatu METAPOST do formatów pdf, eps, jpg, png. • Możliwość przeciągania wierzchołków, z automatycznym przeciąganiem krawędzi. • Koniec trzeciej iteracji, następuje wydanie wersji β i testowanie przy pomocy osób spoza zespołu.
4	1	<ul style="list-style-type: none"> • Możliwość rysowania przez sieć. • Możliwość wklejania tła dla rysunku. • Rozpoczęte prace nad dalszymi funkcjonalnościami opcjonalnymi. • Rozpoczyna się testowanie podstawowej wersji programu (szczególnie aplikacji sieciowej), bez funkcjonalności opcjonalnych, według przygotowanego planu testów.
4	2	<ul style="list-style-type: none"> • Kontynuacja prac nad funkcjonalnościami opcjonalnymi. • Rozpoczynają się testy programu z dodanymi funkcjonalnościami opcjonalnymi, przy pomocy osób spoza Zespołu. • W drugiej połowie czerwca następuje wydanie wersji 1.0 programu GEM i prezentacja tej wersji na zakończenie zajęć ZPP. Zespół rozpowszechnia program na wydziale MIM UW.

4.4. Nadzór i kontrola projektu

4.4.1. Plan zarządzania harmonogramem

Drużyna A jest mało liczną i dosyć zgraną grupą, dlatego nie są konieczne specjalne procedury do pilnowania harmonogramu. W przypadku opóźnień w stosunku do planu interweniować będzie szefowa Zespołu. Postępowanie w przypadku poważnych opóźnień, mogących uniemożliwić powstanie wersji podstawowej projektu na czas, a tym samym zaliczenie zajęć przez Zespół, jest opisane w punkcie 2.4.2.

4.4.2. Plany dotyczące zapewniania i kontroli jakości oraz oceny produktu

Pierwszym zadaniem stojącym przed Drużyną A jest stworzenie dokumentacji. Każdy dokument będzie wielokrotnie czytany przez wszystkich członków Zespołu, którzy będą nanosić poprawki i dyskutować o mniej oczywistych uwagach.

Zostanie uruchomione forum, na którym członkowie Zespołu będą mogli wymieniać wszelkiego typu uwagi dotyczące zarówno dokumentacji, jak i — później — strony programistycznej.

W fazie programowania przewidywane są co najmniej cztery iteracje. Jakość produktu będzie weryfikowana i poprawiana po każdej z nich, za pomocą serii intensywnych testów. W połowie maja przewidywane jest wydanie wersji β , która zostanie udostępniona grupie wybranych wcześniej naukowców i studentów. Zespół będzie starał się uwzględnić wszystkie uwagi zgłoszone przez testerów.

Podczas fazy projektowej powstanie Plan testów, na podstawie którego przy obu wydaniach zostaną przeprowadzone testy programu, podsumowane w odpowiednich raportach, włączonych następnie do dokumentacji projektu.

Po wydaniu wersji 1.0 planowane jest rozpowszechnienie programu. Powstanie specjalne konto e-mailowe, pod które użytkownicy będą mogli przysyłać wszelkiego rodzaju uwagi na temat programu.

4.4.3. Plan raportów

W fazie programowania zespół będzie raz na dwa tygodnie przygotowywał raport z postępów w tworzeniu projektu. Forma raportów zostanie uzgodniona z prowadzącą przedmiot panią Agatą Janowską.

4.5. Plan zarządzania ryzykiem

Zespół będzie dbał o to, by wszelkiego typu awarie sprzętu komputerowego nie zakłóciły powstawania programu. Wykorzystywane będzie dostępne w sieci repozytorium SVN, które będzie często archiwizowane na domowych dyskach członków zespołu.

W przypadku, w którym któryś członek zespołu zrezygnuje z uczestnictwa w projekcie, program w wersji podstawowej zostanie pozbawiony części funkcjonalności. Decyzja o tym, jakich konkretnie, zapadnie, kiedy zaistnieje taka potrzeba, gdyż będzie zależała m.in. od tego, który konkretnie członek Drużyny A odejdzie. Wszelkie tego typu zmiany będą konsultowane z prowadzącą przedmiot.

Procedury na wypadek opóźnień i nieprzewidzanych problemów zostały opisane w punkcie 2.4.

4.6. Plan zamknięcia projektu

Projekt w podstawowej wersji (1.0) zostanie ukończony w czerwcu, razem z końcem zajęć z Zespołowego Projektu Programistycznego. Zespół odda wtedy prowadzącej zajęcia projekt w formie elektronicznej, wraz z repozytorium i kompletną dokumentacją projektową i techniczną. Każdy członek Zespołu dopełni też wtedy formalności związane z licencjatem (podstawą pracy licencjackiej będzie część dokumentacji).

Projekt zostanie przedstawiony szerszemu gronu podczas prezentacji. Prawdopodobnie powstanie też wtedy strona internetowa poświęcona programowi.

5. Plany procesów technicznych

Poniżej zostaną omówione przewidywane techniczne aspekty procesu powstawania projektu.

5.1. Programowanie

Projekt będzie pisany w sposób obiektowy w języku Java przez wszystkich członków Zespołu. Jeśli Zespół znajdzie narzędzie lub bibliotekę udostępniającą którąś z funkcjonalności wymaganych przez produkt, to zostanie rozważone włączenie jej do projektu.

Implementacja projektu będzie odbywać się w czterech iteracjach, w których będą dodawane kolejne funkcjonalności. Po każdej iteracji będzie odbywało się intensywne wyszukiwanie błędów powstałych w trakcie danej fazy projektu i ich naprawianie.

Każda iteracja będzie składała się z dołączenia pewnej liczby modułów. Każdy moduł przed dołączeniem do całego systemu, będzie testowany samodzielnie. W trakcie przyłączania będą testowane interfejsy używane przez dany moduł.

5.2. Metody, narzędzia i stosowane technologie

Program będzie pisany zgodnie z obiektowym paradygmatem programowania.

W trakcie pracy nad projektem Zespół będzie używał następujących technologii i narzędzi:

- System operacyjny z zainstalowanym Java Runtime Environment (JRE w wersji kompatybilnej z 1.5)
- Licencja GPL na której będzie rozpowszechniane oprogramowanie
- Połączenie sieciowe z obsługą protokołu TCP/IP
- XML 1.0
- UTF-8
- PDF
- L^AT_EX, METAPOST i konwertery podstawowych formatów (dystrybucja T_EXLive 2003 lub nowsza)
- Standard zapisu rysunku w XML, stworzony przez Zespół w trakcie pisania projektu
- XML-RPC wersja 3 lub kompatybilna
- Java Swing
- Eclipse IDE
- JavaDoc
- SVN

5.3. Plan akceptacji produktu

Produkt będzie akceptowany przez Zespół po każdej iteracji. W razie problemów, które mogą się pojawić w pewnej iteracji projektu, powstanie szczegółowy plan naprawy błędów, po wdrożeniu którego nastąpi kolejne podejście do zatwierdzenia iteracji.

Końcowa akceptacja produktu będzie wymagać pozytywnej opinii kilku testerów.

6. Plany pomocnicze

6.1. Plan zarządzania zmianami

W razie pojawienia się problemów wymagających znaczących zmian w projekcie, Zespół będzie wspólnie decydował o tym, jakie będą to zmiany. Szczegółowe procedury wprowadzania zmian w projekcie zostały przedstawione w punkcie 2.4.

Pomniejsze zmiany w implementacji konkretnych funkcjonalności może wprowadzić osoba odpowiedzialna za daną funkcjonalność, bez konieczności konsultacji z resztą Zespołu, ale jest zobowiązana bezzwłocznie powiadomić innych o wprowadzonej poprawce.

6.2. Plan dokumentacji

W trakcie pracy nad projektem powstaną następujące dokumenty (data przy dokumentach konkretnie jest datą zajęć ZPP, na które dokument ma być przygotowany):

6.2.1. Dokumentacja podstawowa

Wizja opisująca projekt bardzo ogólnie (30.10.2006),

Plan zarządzania projektem opisujący podział obowiązków między członków Zespołu i wyznaczający ogólne ramy czasowe powstawania projektu (ten dokument, 14.11.2006),

Biznesowe przypadki użycia opisujące wszystkie możliwości projektu z punktu widzenia użytkownika (28.11.2006),

Techniczne przypadki użycia opisują techniczne rozwiązania stojące za wszystkimi przypadkami użycia wymienionymi w „Biznesowych przypadkach użycia” (11.12.2006),

Opis architektury programowej opisujący rozwiązania architektoniczne stosowane w produkcji (9.01.2007),

Plan zatwierdzana opisuje jakie wymagania stawiamy projektowi, aby mógł zostać wydany (20.02.2007),

Plan testów przedstawiający testy, którym zostanie poddany projekt (6.03.2007),

Sprawozdanie z testów i wypuszczenia produktu opisujące wyniki zaplanowanych testów i wypuszczony produkt (wersja β — maj 2007, wersja 1.0 — czerwiec 2007).

6.2.2. Dokumentacja dodatkowa

Opis formatu danych stworzonego przez Zespół do przechowywania wiadomości dotyczących rysunku (styczeń 2007).

Raporty z kolejnych etapów wykonywania projektu w drugim semestrze (krótkie informacje podsumowujące, co udało się wykonać, a czego brakuje w stosunku do planu).

7. Dodatek A — uaktualniony harmonogram prac

7.1. Uwagi ogólne

Uaktualniając plan pracy nad projektem, Zespół postanowił przesunąć pierwszą iterację oraz skrócić drugą. Po rozplanowaniu hierarchii klas programu okazało się, że druga iteracja nie powinna okazać się istotnie bardziej pracochłonna od pozostałych, ponieważ rozwijając opcje graficzne programu, Zespół będzie korzystał z klas zaimplementowanych w pierwszej fazie, które trzeba będzie uzupełnić o metody specyficzne dla różnych rodzajów obiektów graficznych, natomiast mechanizm działania edytora powstanie w pierwszej iteracji. Pierwsza iteracja została przesunięta również dlatego, że obawiamy się, że trudno będzie nam zakończyć pierwszą część projektu przed sesją poprawkową.

Praca nad zadaniami, które według poniższej tabeli mają być wykonywane w dwuosobowych zespołach, może być podzielona dowolnie pomiędzy te osoby, w zależności od możliwości czasowych wykonawców.

W dalszym ciągu tego rozdziału używane są nazwy klas programu, opisane w dokumencie „Projekt architektury systemu”.

7.2. Podział pracy

Iter.	Termin	Zadanie	Wykonawcy
1.	6.03.2007	Implementacja klas zarządzających pracą edytora tekstowego: Edytor, Edytor_Tekstowy, Plik_Tekst.	Maria Donten, Piotr Hofman
1.	6.03.2007	Implementacja klas zarządzających pracą edytora graficznego: Edytor_Graficzny, Rysunek, z podstawowymi metodami. Implementacja klas odpowiedzialnych za wizualizację i interakcję z użytkownikiem — tylko najprostsze funkcjonalności umożliwiające testowanie wyświetlania obiektów.	Kuba Pochrybniak, Marek Grabowski
1.	20.03.2007	Implementacja przeglądarki, działająca możliwość podglądu pliku tekstowego z kodem METAPOST.	Maria Donten, Piotr Hofman
1.	20.03.2007	Implementacja klas obsługi grafiki — klasy Obiekt_Graficzny i jej podklas, bez krawędzi, etykiet i siatek.	Kuba Pochrybniak, Marek Grabowski
2.	3.04.2007	Implementacja klasy Etykieta, interfejs dodawania etykiety do rysunku.	Maria Donten
2.	3.04.2007	Krzywe Béziera — interfejs dodawania krzywych do rysunku.	Kuba Pochrybniak
2.	3.04.2007	Menedżer obiektów — implementacja klasy Menedżer, zaznaczanie zbiorów obiektów za pomocą menedżera.	Piotr Hofman
2.	3.04.2007	Pomniejszanie, powiększanie i przesuwanie obszaru widocznego na szkicowym rysunku, zaznaczanie zbiorów obiektów na rysunku.	Marek Grabowski

2.	17.04.2007	Siatki — implementacja klasy Siatka, możliwość włączania/wyłączania siatki na rysunku, algorytm wyszukiwania punktu siatki najbliższego do punktu o danych współrzędnych.	Maria Donten
2.	17.04.2007	Krzywe Béziera — zmiana kształtu krzywych, dodawanie punktów do krzywych.	Kuba Pochrybniak
2.	17.04.2007	Menedżer obiektów — sterowanie kolejnością wyświetlania obiektów za pomocą menedżera.	Piotr Hofman
2.	17.04.2007	Uzupełnienia własności obiektów graficznych i edytora (kolor obiektu, wycinanie i wklejanie...)	Marek Grabowski
3.	1.05.2007	Obsługa rysowania grafów — podstawowe własności klasy Krawędź.	Piotr Hofman
3.	1.05.2007	Konfiguracja programu — implementacja klasy Konfig, włączenie możliwości zmian konfiguracji do programu.	Maria Donten
3.	1.05.2007	Możliwość cofania zmian na rysunku, implementacja klasy Stosy.	Kuba Pochrybniak
3.	1.05.2007	Kompilacja języka METAPOST i konwersja formatów — implementacja klasy Komp_Konw.	Marek Grabowski
3.	15.05.2007	Tłumaczenie wewnętrznego obiektowego formatu rysunku na kod METAPOST (klasa Grafika2MP).	Maria Donten, Kuba Pochrybniak
3.	15.05.2007	Podłączenie do programu zewnętrznych konwerterów formatów i kompilatora języka METAPOST za pomocą klasy Komp_Konw.	Marek Grabowski
3.	15.05.2007	Implementacja operacji na krawędziach grafu — zmiana wierzchołków, zmiana przebiegu.	Piotr Hofman
4.	29.05.2007	Aplikacja sieciowa — klasy Klient i Serwer.	Marek Grabowski, Maria Donten
4.	29.05.2007	Możliwość dodawania do rysunku tła wczytanego z pliku graficznego.	Kuba Pochrybniak
4.	29.05.2007	(opcjonalnie) Przekształcenia afiniczne na zbiorach obiektów rysunku.	Piotr Hofman
4.	12.06.2007	(opcjonalnie) Przekształcenia afiniczne na zbiorach obiektów rysunku — kontynuacja prac.	Piotr Hofman
4.	12.06.2007	(opcjonalnie) Znajdowanie punktów przecięcia krzywych.	Kuba Pochrybniak
4.	12.06.2007	Ewentualne uzupełnienia brakujących funkcjonalności.	Marek Grabowski, Maria Donten

8. Historia zmian

Wersja	Data	Autorzy zmian	Zmiany
0.5	7.11.2006	cały Zespół	pierwotna, niepełna wersja
0.6	7.11.2006	Marek Grabowski	dodany pierwszy rozdział, uzupełnienie planów
0.7	8.11.2006	Maria Donten	poprawki merytoryczne oraz językowe, uporządkowanie dokumentu, zmiany w harmonogramach
0.8	9.11.2006	Kuba Pochrybniak	uzupenienie brakujących dotąd planów
0.9	10.11.2006	Maria Donten	szczegółowy harmonogram pracy, poprawki w teście
0.95	11.11.2006	Kuba Pochrybniak	poprawki merytoryczne oraz językowe
1.0	11.11.2006	Marek Grabowski	drobne poprawki
1.1	1.12.2006	Kuba Pochrybniak	poprawki (uwzględnione uwagi prowadzącej)
1.2	15.02.2007	Maria Donten	aktualizacja harmonogramu — Dodatek A