Predykcyjno-przyrostowa strategia rozwoju informatycznych aplikacji dedykowanych*

Henryk Krawczyk
Rafał Knopa, Sebastian Kruk
Adam Mazurkiewicz, Jan Zieliński
Katedra Architektury Systemów Komputerowych
Wydział Elektroniki, Telekomunikacji i Informatyki
Politechnika Gdańska

Streszczenie

Przedstawiono doświadczenie związane z budową i rozbudową trzech kolejnych wersji aplikacji informatycznej przeznaczonej dla zastosowań medycznych. Scharakteryzowano przyrostowy model rozwoju aplikacji i skomentowano jego przydatność przy opracowaniu kolejnych wersji na podstawie prognozowanych wymagań. Opisano również finalną architekturę aplikacji jak i jej interfejs z użytkownikiem.

1. Wprowadzenie

Wytwarzanie oprogramowania jest działalnością intelektualną wymagającą różnorakiej wiedzy, w tym co najmniej pochodzącej z dwóch różnych obszarów takich jak:

1. metodyki i narzędzia wytwarzania oprogramowania,
2. specyfiki dziedziny rozwiązywanego problemu.

*Praca zrealizowana w ramach grantu KBN 8T11C00117

W wielu podejściach zakłada się, że metodyka wytwarzania oprogramowania jest na tyle uniwersalna, że można ją wykorzystać do budowy każdej aplikacji [2], [8]. Wyróżnia się kilka modeli cyklu życia (kaskadowy, V, czy spiralny) i precyzuje się zadania dla każdego z etapów całego procesu wytwarzania. Zakłada się, że idealne ich wykonanie prowadzi do sukcesu, co oznacza wytworzenie produktu o oczekiwanych przez użytkownika własnościach. Z praktyki informatycznej wynika jednak, że specyfika dziedziny aplikacji wymaga często podejścia zorientowanego na tę dziedzinę. Stąd też coraz popularniejsze stają się różnego typu wzorce projektowe [10], określone na podstawie konkretnych aplikacji, które są często wykorzystywane w danej dziedzinie i które uzyskały szeroką akceptację. Z uwagi na częstą i szybką zmianę technologii, a także modyfikację wymagań użytkownika, mniej istotne staje się dopasowanie architektury systemu do konkretnych (często chwilowych) wymagań, a raczej konstrukcja aplikacji o dużej elastyczności, w znacznie krótszym czasie, które łatwo poddają się potem różnego typu modyfikacjom.

Ogólne rozwiązania wychodzące na przeciw tym ostatnim tendencjom sprowadzają się do modyfikacji poprzednio opracowanych modeli cyklu życia. Przykładem jest strategia małych zespołów realizujących równolegle swoje zadania w sposób iteracyjny lub metodą przyrostową (incremental or iterative product development) [4]. Innym podejściem jest rozwój wielu kolejnych wersji, metodą ewolucyjną (version staged model) [9]. Jeszcze innym znanym podejściem tzw. negocjacyjnym jest rozbudowa modelu spiralnego (WinWin spiral model) [3] o dodatkowe fazy, co umożliwia lepszą koncentrację na realizacji prostszych zadań. Przyrostowe podejście do budowy aplikacji oznacza często składanie jej z kolejno oprowacowywanych małych komponentów w coraz to większą konstrukcję. Złożone aplikacje, integrujące pracę różnych istniejących już systemów podano w pracy [1]. W tym przypadku szczególną uwagę zwraca się na różny sposób postrzegania aplikacji przez udziałowców, klientów, architektów, wykonawców i osoby odpowiedzialne za utrzymanie. Dwa pierwsze wyżej wymienione podejścia dotyczą raczej aplikacji powszechnego stosowania, zaś dwa ostatnie związane są z budową aplikacji dedykowanych, zorientowanych na konkretne dziedziny zastosowań, takie jak aplikacje multimedialne, czy handel elektroniczny.

Przedstawiono poniżej metodą i budowę komputerowego systemu medycznego wspomagającego rejestrację badań endoskopowych i wspomaganie rozpoznawania chorób gastroenterologicznych. Taki system (o nazwie ERS [6] – Endoscopy Recommender System) jest rozwijany w Katedrze Architektury Systemów Komputerowych od 1997 roku przy współpracy Kliniki Gastroenterologicznej Akademii Medycznej w Gdańsku. W rozdziale drugim przedstawiono sposób rozwoju kolejnych wersji systemu, przy wykorzystaniu zarówno modelu wersjowego i przyrostowego oraz założeniu predykcji zmian wymagań. W rozdziale następnym opisano wersję finalną architektury systemu i interfesju użytkownika. We wnioskach końcowych zwrócono uwagę na możliwość dalszej rozbudowy systemu oraz podsumowano wykorzystaną metodę wytwarzania.

2. Hybrydowe podejście do budowy oprogramowania

Projektowana aplikacja dla zastosowań medycznych jest aplikacją dedykowaną, gdzie przyszły użytkownik (lekarz) nie jest specjalistą z technologii informatycznej, dlatego trudno jest mu przedstawić szczegółowe wymagania na tego typu aplikację. Spodziewa się on podstawowych korzyści: odciążenia go od pracy administracyjnej, w tym pomoc przy wypełnianiu dokumentów badań oraz sporządzania dodatkowych sprawozdań. Z uwagi na to, że odbiorcą systemu jest Klinika Akademii Medycznej, wspomina się również o pewnych funkcjach dydaktyczno-szkoleniowych. W miarę jednak eksploatacji systemu, użytkownik zauważa szczegółowe problemy do rozwiązania, na przykład zapewnienie wspomagania pewnych zadań badawczych, czy współpracę tego typu systemów dla wielu klinik. Dzięki temu cały projekt jak i aplikacja ulegają z czasem całkowitej przebudowie.

W tablicy 1 przedstawiono opracowane warianty aplikacji ERS. Pierwsza wersja była realizowana w tradycyjnym podejściu związanym z modelem kaskadowym. Następne zostały zrealizowane po dwuletniej eksploatacji pierwszej wersji bazując na modelu spiralnym, przy czym wykorzystano podejście predykcyjno-przyrostowe. Oznacza ono przyjęcie stopniowego rozwoju aplikacji, gdzie kolejno rozbudowuje się jej komponenty w oparcji o dobrze określone fragmenty funkcjonalne (tzw. przyrosty), zaś wizję wymagań dla tych fragmentów opracowuje się równolegle do istniejącej już wersji.

Na rys. 1 przedstawiono uproszczony schemat postępowania. Rozwój nowej wersji następuje równolegle po przekazaniu do eksploatacji wersji poprzedniej, zaś wersja bieżąca powinna być ukończona przed wycofaniem z pracy wersji poprzedniej. W przedstawionym przypadku budowa każdej nowej wersji o znacznie lepszych możliwościach funkcjonalnych realizowana była znacznie krócej, mimo, że zespoły ją realizące były inne. Podstawą takiego podejścia jest nie tylko zmiana technologii informacyjnej (z DOS + Clipper na Javę), ale również przyjęcie elastycznej architektury aplikacji. Złożona jest ona z dobrze określonych komponentów, które przy założonych odpowiednich interfejsach mogą być niezależnie rozwijane. W wyniku takiego podejścia powstała nie tylko wymagana aplikacja, ale produkty dodatkowe (jak atlas wzorcowych zdjęć endoskopowych czy równoległe środowisko do analizy obrazów medycznych), których funkcjonalność przekracza znacznie oczekiwane możliwości. Co istotne koszt i harmonogram realizacji tego projektu nie był przekraczony.

Tablica 1: Charakterysytka opracowanych wersji aplikacji ERS

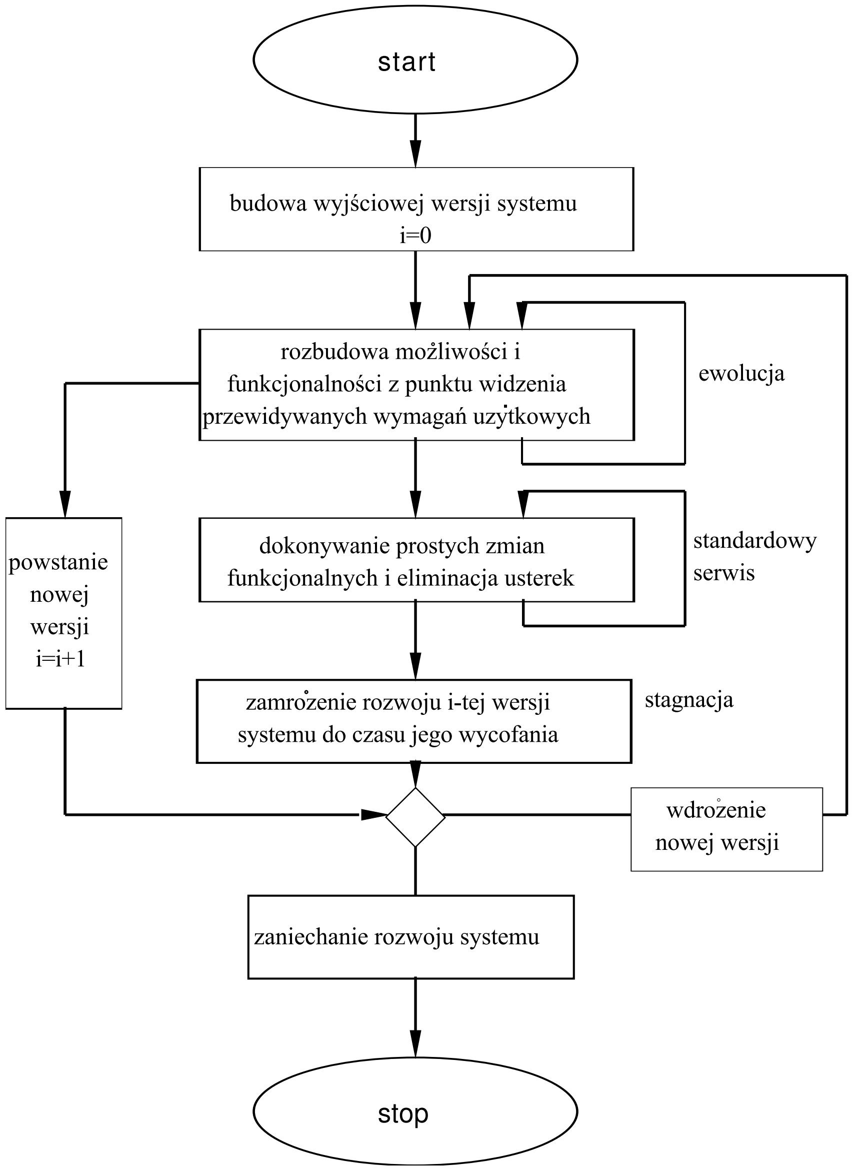
Wersja, rok i czas prod.	Funkcje	Platforma implement.	Koncepcja intefejsu	Uwagi
1, 1997, 2 lata	rejestracja danych pacjenta, rejestracja wyników dla 6 rodz. badań, przegląd i modyfikacja danych, tworzenie i drukowanie raportów i statystyk	MS-DOS, Clipper	ukierunkowany na wyodrębnione typy badań; każda akcja ma ściśle określony scenariusz	system zamknięty funkcjonalnie
2, 1999, 1 rok	rejestracja i obróbka zdjęć endoskopowych, konwersja danych do formatu DICOM, porównywanie zdjęć w oparciu o histogramy	MsWindows 98, Java JDK1.2, MsSQL 7.0	ukierunkowany na dowolne badanie pacjenta; możliwość obróbki filmu i zdjęć z badania; koncepcja prostych pluginów porównywania obrazów; rekomendacja zdjęć oparta na jednym algorytmie	zmiana technologii, wykorzystanie systemu dwukomputerowego
3, 2001, 6 m-cy	standaryzacja opisu badania wg MST, mechanizm rekomendacji wykorzystujący równoległe algorytmy rozpoznawania obrazów, definicja interfejsów komunikacji przez Internet, mechanizm opisu zdjęć wzorcowych	MsWindows 98/2000, Java JDK1.3, JPVM, PVM, RMI, MySQL 3.23	baza wzorców dostępna w Internecie; rozwinięcie technologii pluginów i zaawansowania rekomendacji; wiele funkcji zespolonych z technologiami Internetowymi (np. zestawienia, raporty)	wyodrębnienie dobrze zdefiniowanych komponentów, sieciowe środowisko zintegrowane i rozproszone

3. Architektura systemu ERS

System ERS słada się z wielu komponentów. Na rys. 2 przedstawiono schemat architektury systemu. W kolejnych podrozdziałach opisano wybrane funkcje komponentów systemu.

3.1. Przechwytywanie obrazów

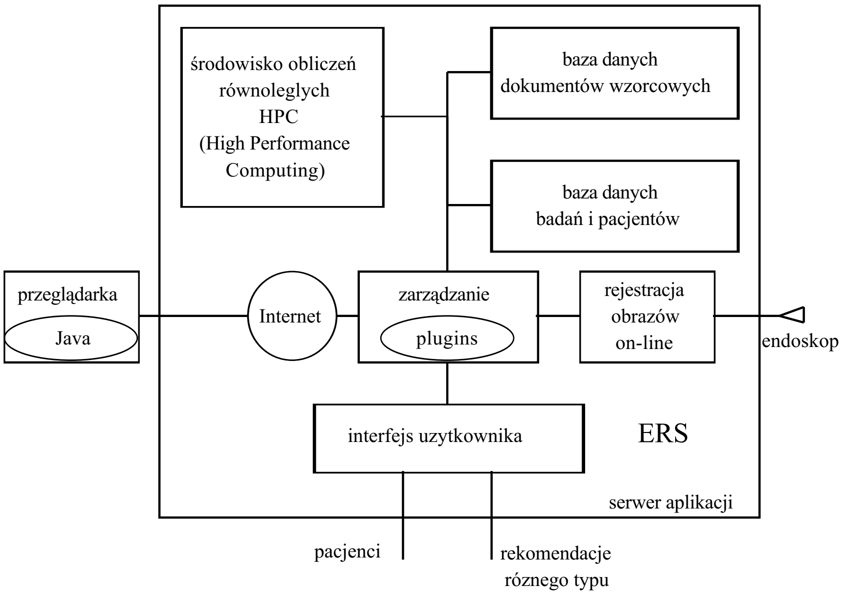
Pacjent poddawany jest badaniu za pomocą endoskopu. Dane z tego urządzenia przesyłane są poprzez złącze video karty graficznej komputera. Proces przechwytywania kontrolowany jest przez klasy pakietu `capture`. Te z kolei wykorzystywane są przez obiekty dostępu do bazy danych (agentów baz danych) w celu zapisywania i odczytywania danych medialnych. Do bazy trafiają także podstawowe dane o pacjencie a także informacje pozyskane w trakcie wywiadu. Takie rozwiązanie umożliwia dostosowanie struktury bazy danych do indywidualnych potrzeb każdej implementacji systemu bez potrzeby ingerencji w kod pakietu `capture`. Po przeprowadzeniu badania pacjenta i ewentualnej rekomendacji co do rozpoznania, zapisy w bazie uzupełniane są szczegółowym opisem badania, zgodnym z międzynarodową normą MST i standardem DI-



Rysunek 1: Koncepcja modelu predykcyjno-przyrostowego rozwoju oprogramowania

COM i wynikami procesu rekomendacyjnego określającego stan pacjenta.

Proces przechwytywania obrazów łączy w sobie zagadnienia z dziedziny optyki, elektroniki i informatyki. Dodatkowo, komplikuje go fakt wielkiej różnorodności rozwiązań sprzętowych i programowych. Jeżeli na dodatek odbywać ma się on w czasie rzeczywistym, to trzeba zaangażować całkiem spore zasoby sprzętowe. Dodatkowym założeniem ERS była niezależność od platformy, co uzyskano programując całość systemu w języku Java, wykorzystując



Rysunek 2: Architektura systemu ERS

pakiet JMF.

Program napisany w Javie, obojętnie czy jest to aplikacja, aplet, czy też komponent *JavaBeans*, korzysta z funkcji udostępnianych przez warstwę przetwarzania i prezentacji pakietu JMF. Klasy pakietu `caputre` dostarczają niezbędnego do przeprowadzenia przechywytywania interfejsu użytkownika oraz ukrywają przed programistami innych częsci systemu szczegóły dostępu do klas JMF. Aby zapewnić odpowiednią wydajność procesu przechwytywania, niezbędnym okazało się również użycie `Windows Performance Pack`.

3.2. Bazy danych

Dane przechowywane przez system ERS dzielą się na dwie grupy: dane jawne, czyli baza danych obrazów i opisów wzorcowych wraz z ich opisem, oraz poufne, takie jak dane personalne pacjenta, wyniki przeprowadzonego z nim wywiadu, wyniki poszczególnych badań, zlecająca je instytucja medyczna, dane personalne lekarza badającego, a także wyniki rekomendacji. Dostęp do bazy danych może odbywać się za pośrednictwem agentów dostępu do baz danych, co daje większą elastyczność w doborze motoru (engine'u) bazy danych, lub bezpośrednio – poprzez odwołania do `JDBC (Java Data Base Connectivity)`.

Dane dotyczące opisu chorób są ściśle związane ze standardem MST, który stanowią wzorce złożone z reprezentatywnych zdjęć i ich opisów. Schemat bazy danych jest trójpoziomowy. Najwyżej znajdują się tabele główne: zdjęcia i nagłówki dokumentów MST. Poniżej zaś tabele reprezentujące sekcje MST.

Każdej sekcji odpowiada oddzielna tabela. Tabele sekcji zawierają kolumnę z identyfikatorem dokumentu, która wiąże dokument MST z zawartościami poszczególnych sekcji. Na najniższym poziomie znajdują się tabele słownikowe. Tabele słownikowe pozwalają uniezależnić tabele sekcji od opisów słownych. Istnieją trzy tabele słownikowe:

- tabela słów ze standardu MST – do niej są dopisywane słowa pochodzące z plików konfiguracyjnych MST,
- tabela opisów – zawiera opisy z części MST pozwalających na wprowadzanie swobodnych opisów,
- tabela jednostek chorobowych – zawiera określenia dla rozpoznań końcowych.

Ponieważ w tabelach sekcji dokumentów MST nie są umieszczane żadne opisy słowne, lecz identyfikatory do pól w tabelach słownikowych, tylko tabele słownikowe mogą zawierać opisy słowne. Takie podejście ułatwia przetwarzanie automatyczne, jak np. tłumaczenie na inne języki, porównywanie słów itd. Ponadto umożliwia to przyspieszenie dostępu do bazy danych (np. poprzez buforowanie w pamięci słowników) oraz upraszcza procedury korekty błędnie wprowadzonych globalnych sformułowań (np. błędów w plikach konfiguracyjnych).

Korzystanie z tak zbudowanej bazy danych jest wydajne. W tabelach prostych wymaga tylko pojedynczego odwołania się do tabeli. W tabelach złożonych pierwsze odwołanie pozwala uzyskać liczbę grup określeń, aby w kolejnych odwołaniach już uzyskiwać konkretne informacje. W przypadku klasycznej relacyjnej bazy danych należałoby wykonywać kolejne zapytania do oddzielnych tabel stworzonych dla poszczególnych określeń. Drugą niekwestionowaną zaletą jest to, że modyfikowacje struktury MST nie wpływają na strukturę bazy danych. Dzieje się tak dlatego, że struktura dokumentu MST nie jest wpisana w strukturę bazy danych, lecz stanowi część danych zapisanych w bazie danych

3.3. Procedury rekomendacji

Dotyczy on wydania diagnozy i podjęcia decyzji o dalszym leczeniu badanego pacjenta. Z uwagi na złożoność operacji porównywania dokumentów może zostać wykorzystane środowisko równoległego przetwarzania. Wywiad z pacjentem określający występujące objawy chorobowe stanowi pierwszą część rekomendacji, tzn. podstawę określania rozpoznania wstępnego. Na podstawie informacji zawartych w karcie badania pacjenta, w części dotyczącej objawów, system ERS stara się znaleźć kilka najbardziej podobnych pod względem objawów dokumentów z bazy dokumentów wzorcowych. Zaproponowane przez system dokumenty wzorcowe mogą posłużyć lekarzowi w celu ustalenia prawdopodobnej choroby pacjenta. Informacje te mogą okazać się pomocne także do ustalenia dalszych badań pacjenta.

Po wykonaniu badania endoskopowego, system ERS w trzech krokach rekomenduje wypełnienia pozostałej części karty badania pacjenta i ustalenia ostatecznej diagnozy. Pierwszy krok dotyczy rekomendacji jedynie na podstawie zdjęć endoskopowych wykonanych podczas badania. Drugi krok korzysta dodatkowo z informacji (częściowo wypełnionego) dokumentu pacjenta. W ostatnim kroku selekcjonowane są informacje z kroków poprzednich oraz sugestie lekarza.

W pierwszym kroku system ERS dokonuje porównania zdjęcia (zdjęć) z badania endoskopowego ze zdjęciami wzorcowymi. W wyniku porównania przedstawiane zostaje lekarzowi maksymalnie pięć dokumentów wzorcowych zawierających zdjęcia najbardziej zbliżonych do zdjęć pochodzących z danego badania endoskopowego. Jeżeli lekarz dojdzie do wniosku, że jeden z zaprezentowanych dokumentów w sposób dostatecznie zbliżony opisuje obecne badanie, wówczas może on przepisać zawarte w nim informacje do karty badania pacjenta i w tym momencie zakończyć proces rekomendacji. Jeżeli żaden z dokumentów nie opisuje dostatecznie dobrze obecnego badania, wówczas konieczne jest wypełnienie, choćby fragmentarycznie, karty badania pacjenta. Wówczas w drugim kroku rekomendacji system stara się zaprezentować maksymalnie pięć dokumentów o zawartości zbliżonej do tej, którą lekarz już umieścił w karcie badania oraz o zdjęciach zbliżonych do zdjęć znajdujących się w dokumentach wzorcowych. Podobnie jak poprzednio, lekarz ma wówczas możliwość wyboru jednego z dokumentów w celu wypełnienia zawartości karty badania pacjenta, i zakończenia procesu rekomendacji. W ostatnim kroku system dokonuje porównania zwróconych wcześniej dokumentów wzorcowych i prezentuje lekarzowi maksymalnie dwa, które najlepiej pasują do obecnego badania. Prezentowane dokumenty są ostatnimi proponowanymi przez system ERS. Na ich podstawie lekarz ma możliwość wypełnienia karty badania. Krok ten kończy jednocześnie proces rekomendacji. W momencie wyboru jednego z prezentowanych dokumentów wzorcowych jako referencyjnego do wypełnienia karty badania, system podaje również sugerowane procedury diagnostyczne i terapeutyczne oraz prawdopodobne powikłania, które mogą wystąpić.

3.4. Rozwój interfejsu

W trakcie kolejnych faz rozwoju systemu funkcjonalność (p. tablica 1) i wygląd interfejsu zmieniały się z uwagi na wykorzystanie nowych funkcji systemu, czy zmiany systemu operacyjnego i platformy programistycznej. Początkowy kształt interfejsu opierał się na dość ścisłym powiązaniu pól tekstowych i innych elementów operacji użytkownika ze strukturą bazy danych. Rejestracja obrazów z endoskopu do tej wersji systemu została dodana dużo później jako zupełnie osobny moduł i w praktyce nie była wykorzystywana w systemie. W kolejnej wersji, już zorientowanej na kompleksową rejestrację badania, funkcje związane z przechwytywaniem i szeroko rozumianymi modyfikacjami obrazu stały się integralną, ważną częścią systemu, pełniąc rolę elementu scenariusza tworzenia rekordu pacjenta. Rekord pacjenta jest tutaj koncepcją logiczną, ktora wiąże wszelkie dane związane z konkretnym pacjentem w jedną sensowną całość. Ostatecznie koncepcja interfejsu użytkownika systemu ERS opiera się na przyjaznym przeprowadzeniu operatora systemu przez wszystkie etapy tworzenia i zarządzania rekordem pacjenta. Główną nowością tej wersji jest wykorzystanie w opisie badania standardu MST oraz możliwości dokonywania rekomendacji w dowolnym momencie tworzenia opisu badania.

Hierarchiczna struktura standardu MST umożliwiła stworzenie przyjaznego interfejsu dla rejestracji wyników konkretnego badania. Przy tym wyróżniono dwie części: pierwsza część, nadrzędna pokazuje aktualny wygląd dokumentu w formie gotowego do wydruku arkusza i, za pomocą odnośników kodu HTML, modyfikuje dynamicznie jego zawartość, wołając w razie potrzeby metody części drugiej. Druga część składa się z okienek dialogowych, które pozwalają podawać wartości dla kolejnych pól standardu MST i jednocześnie prowadzą operatora przez niejednoznaczności, które mogą się pojawić dla danego dokumentu.

4. Wnioski końcowe

W wyniku zastosowania opisanej metody wytwarzania powstał całkiem złożony system o dużej funkcjonalności z możliwością prostej rozbudowy. Dotyczy to zwłaszcza efektywnych procedur porównywania obrazów, czy skutecznych procedur rekomendacji, które mogą być iteracyjnie udoskonalanie. Bardzo istotne jest także określenie wielkości komponentów, z których budowana jest aplikacja dedykowana oraz możliwych trendów ich rozbudowy. Niezbędna jest przy tym duża znajomość platformy i dziedziny zastosowań, w celu oszacowania które ze zmian są najbardziej przydatne i łatwe do zrealizowania. To doprowadziło do rozwoju kilku mniejszych aplikacji, jak atlas zdjęć wzorcowych (wzorcowa baza danych), środowisko równoległej analizy obrazów [7], czy pakiet rejestracji pacjentów (wbudowany w interfejs użytkownika ERS), które zostały najpierw szczegółowo przetestowane, by stanowić wbudowaną cegiełkę przy konstrukcji systemu ERS.

Skuteczność zaproponowanej metody predykcyjno-przyrostowego wytwarzania zależy nie tylko od wykorzystywanej technologii, odpowiedniej predycji wymagań, ale również od oceny jakościowej powstałych produktów i procesów im towarzyszących. Istnieje możliwość wykorzystania tutaj pakietu SOJO [5], który zarówno wspomaga proces oceny jakości jak i daje możliwość jego prognozowania. Precyzja oceny i prognozowania jakości ma istotny wpływ na liczbę iteracji (p. rys. 1) jak i na decyzję podjęcia budowy nowej wersji systemu.

Literatura

[1] F. J. Armour, S. H. Kaisler, and S. Y. Liu. Building an enterprise architecture step by step. *IT Pro*, July/August:31–38, 1999.

[2] A. Behforaoz and F.J. Hudson. Software engineering fundamentals. Oxford University Press, New York, 1996.

[3] B. Boehm, A. Egyee, J. Kwan, D. Post, A. Shoh, and R. Madochy. Using the winwin spiral model: A case study. *IEEE Computer*, July:33–44, 1998.

[4] M.A. Cusumono and R.W. Selby. How microsoft builds software. *Comm. of the ACM*, June:53–61, 1997.

[5] P. Czarnul, H. Krawczyk, and A. Mazurkiewicz. Quality driven development methodology for network applications. In *Proceedings of the ISThmus Conference*. 2000.

[6] H. Krawczyk and R. Knopa. Web-based endoscopy recommender system – ers. In *Proc. of IEEE Conference on Parallel Computing in Elec. Engineering*, pages 257–261. Trois Rivieres, Quebec, Canada, 2000.

[7] H. Krawczyk, P. Nowikowski, and A. Neyman. Parallel environment for endoscopy image analysis. *Praca ukaże się w Lecture Notes of Computer Science*, 2001.

[8] H. Krawczyk and B. Wiszniewski. *Analysis and Testing Distributed Applications.* RSP Ltd, Baldock, Hertfordshire, England, 1998.

[9] V.T. Rajlich and K.H. Bennett. A staged model for the software life cycle. *IEEE Computer*, July:66–71, 2000.

[10] S. Svinivasan. Design patterns in object-oriented framewoks. *IEEE Computer*, Feb.:24–31, 1999.