Paweł Gesek

**Automatyczna budowa modelu na podstawie specyfikacji wymagań**

1. **Wstęp**

Specyfikacja Wymagań Systemowych(SWS) jest jednym z kluczowych elementów procesu wytwarzania oprogramowania. Błędy popełnione w fazie pisania specyfikacji w sposób nieunikniony dadzą o sobie znać w późniejszych fazach tworzenia systemu.

Wymagania pisane są w języku naturalnym, co może prowadzić do niejasności, nieporozumień itp. Również automatyczne wyciąganie semantyki z języka naturalnego jest zadaniem trudnym, co jest główną przeszkodą w automatyzacji procesu przetwarzania specyfikacji wymagań oraz budowy modelu klas na jej podstawie. Powoduje to, że obecnie większość narzędzi komercyjnych nie oferuje możliwości automatycznej analizy specyfikacji wymagań w języku naturalnym. Zaproponowanych zostało kilka metod starających się rozwiązać ten problem.

1. **Automatyczne przejście z SRS do prezentacji formalnej**

Jedna z przedstawionych w literaturze metod [1] skupia się na automatycznym tłumaczeniu języka naturalnego zawartego w specyfikacji wymagań na zorientowany obiektowo model analityczny. Metodologia ta nie nakłada żadnych restrykcji na język naturalny specyfikacji, co prowadziłoby do znacznego ograniczenia jej użyteczności, co mija się z celem. Zamiast tego, stara się imitować ludzki proces analizy, który dzieli problem na części w celu zmniejszenia jego złożoności.

Metodologię tą można podzielić na trzy części: a) komponent lingwistyczny; b) sieć Semantyczną , która tworzy formalną reprezentację języka naturalnego; c) modelowanie obiektowe, ostatnią fazę, w której z sieci semantycznej powstaje obiektowy model analityczny.

Zadaniem komponentu lingwistycznego jest organizowanie części mowy występujących w specyfikacji i organizowanie ich ze względu na rolę pełnioną w zdaniu w trzy grupy: podmiot, orzeczenie, dopełnienie. Te grupy tworzą zdania, które z kolei mogą tworzyć zdania złożone. Można więc wyróżnić trzy poziomy: 1) słowa tworzą grupy; 2) grupy tworzą zdania; 3) zdania tworzą zdania złożone. Ta wiedza wydobyta z języka naturalnego pozwala stworzyć sieć semantyczną , która daje pełny obraz związków pomiędzy obiektami (rzeczownikami – podmiotami lub dopełnieniami). Ta wiedza w postaci sieci semantycznej może być łatwo przetłumaczona na model obiektowy.

1. **Metodologia oparta na Rational Unified Process (RUP)**

Kolejna metodologia stara się analizować SRS w sposób zbliżony do działań podejmowanych w dwóch pierwszych fazach RUP[2]. Na początku tworzony jest model przypadków użycia. Kandydaci na aktorów wybierani są spośród rzeczowników występujących w specyfikacji wymagań, natomiast przypadki z czasowników. Odpowiednie heurystyki mogą być wykorzystane w celu polepszenia jakości zbioru aktorów i przypadków użycia.

Kolejnym krokiem jest zbudowanie modelu klas na bazie wygenerowanego wcześniej zbioru przypadków użycia. Na początku należy wytypować klasy i zidentyfikować ich stereotypy orz wygenerować diagram klas analitycznych. Następnie należy rozłożyć zachowania systemu na czynniki pierwsze i rozdzielić je między wcześniej wytypowane klasy. Konieczne jest też wyłuskanie związków pomiędzy klasami i przełożenie ich na asocjacje lub generalizacje. Ważnym elementem generacji modelu jest słownik opisujący domenę. Jeśli jednostka z przypadków użycia pojawia się w słowniku, staje się silnym kandydatem do stania się klasą w modelu analitycznym. Do generacji modelu klas wykorzystywany jest również wcześniej zdefiniowany zestaw zasad. W wygenerowanym modelu wyróżnić można: a) klasy graniczne, przedstawiające interfejs systemu; b) klasy reprezentujące obiekty świata rzeczywistego; c) klasy kontrolne odpowiedzialne za przepływ wydarzeń. Klasy kontrolne są zależne do aplikacji i ich wytypowanie jest sprawą bardzo subiektywną.

Ostatnim krokiem jest walidacja wygenerowanego modelu, polegająca na sprawdzeniu czy nie pojawiają się niedozwolone związki np. pomiędzy dwoma aktorami lub aktorem i obiektem kontrolnym.

1. **Minimalizowanie wieloznaczności w języku naturalnym w SRS.**

Innym podejściem jest przedstawienie kontrolowanej reprezentacji języka naturalnego dla wymagań systemowych , z którego można łatwiej wygenerować dobrej jakości modele klas. Aby było to możliwe, konieczna jest automatyczna metodologia pozwalająca generować tą kontrolowaną reprezentację z języka naturalnego. W literaturze można znaleźć przedstawienie metody generującej of Business Vocabulary and Rules (SBVR) [3]. SBVR jest standardem Object Managment Group(OMG) [4]. Dzięki SBVR możliwe staje się uchwycenie wymagań systemowych w kontrolowanym języku, który może być znacznie łatwiej analizowany przez maszynę.

Typowa reprezentacja wymagań w SBVR składa się ze słownika oraz zasad biznesowych. Słownik zawiera wszystkie specyficzne definicje konceptów i terminy używane przez daną organizację. W SBVR koncept może być konceptem rzeczownikowym lub konceptem czasownikowym(faktem). Koncepty rzeczownikowe mogą być dać podzielone na typy obiektów, koncepty indywidualne oraz charakterystyki, natomiast koncept czasownikowy przedstawia związek między konceptami czasownikowymi. Zasady SBVR opisują natomiast zachowania jednostek biznesowych oraz strukturę organizacji.

Tłumaczenie języka naturalnego do SBVR zaczyna się od przetwarzania leksykalnego. W tej fazie zdania rozbijane są na słowa, które identyfikowane są jako poszczególne części mowy przez parser Stanford. W następnej fazie parser analizuje tekst syntetycznie, generując drzewo semantyczne. Kolejna faza polega na interpretacji semantycznej. W jej trakcie przypisywane są role takie jak: aktor(rzeczownik), ko-aktor(aktorzy złączeni spójnikiem), akcje(czasownik), obiekty tematyczne(rzeczowniki używane w dopełnieniu) i beneficjentów(czasownik użyty w dopełnieniu). Następnie, przy użyciu grupy zasad, tworzony jest słownik SBVR poprzez identyfikowanie konceptów z obrobionego w poprzedniej fazie tekstu. W ostatniej fazie tworzone są zasady SBRV, które podobnie jak słownik generowane są przy użyciu wcześniej wyznaczonych zasad.

1. **Bibliografia**
2. M.G. Ilieva, Olga Ormandjieva , Automatic Transition of Natural Language Software Requirements Specification into Formal Presentation
3. D. Liu, K. Subramaniam, A. Eberlein, B.H. Far, Natural Language Requirements Analysis and Class Model Generation Using UCDA
4. A. Umber, I.S. Bajwa, Minimizing Ambiguity in Natural Language Software Requirements Specification
5. Object Managment Ggroup, Semantics of Buisness vocabulary and Rules, Standard v.1.0, dostępne: <http://www.omg.org/spec/SBVR/1.0/>