

Protokół aukcji internetowych

Piotr Kufel

1 maja 2009

Spis treści

1	Streszczenie	3
2	Terminologia	3
3	Cele protokołu	3
4	Założenia	3
5	Format komunikatów	4
6	Opis komunikatów	4
6.1	Specjalne komunikaty	4
6.2	Komunikaty fazy I	5
6.3	Komunikaty fazy II	6
7	Sytuacje wyjątkowe	8
8	Opis komunikacji	9
8.1	Negocjacja wersji	9
8.2	Faza I	9
8.3	Faza II	10
9	Wykaz stałych	13
9.1	Komunikaty	13
9.2	Kody błędów dla ACK_MSG	13
9.3	Inne stałe	13

1 Streszczenie

Niniejszy dokument opisuje protokół obsługi aukcji internetowych w wersji 1.

2 Terminologia

Serwer Instancja protokołu udostępniająca aukcję, strona pasywna.

Klient Instancja protokołu łącząca się z serwerem, strona aktywna.

Sprzedawca Osoba korzystająca z klienta.

Kupujący Osoba korzystająca z serwera.

Niepoprawny komunikat Komunikat, który został odebrany w nieprzewidzianym przez specyfikację momencie, lub przynajmniej jedno jego pole nie spełnia podanych w specyfikacji własności.

Aukcja Sesja w której uczestniczy dokładnie jeden sprzedawca i dowolna liczba klientów. Dotyczy wszystkich przedmiotów wystawianych przez sprzedawcę. Jest jednoznacznie wyznaczona przez adres serwera.

Licytacja Dotyczy pojedynczego przedmiotu dostępnego na aukcji.

I faza Okres, w którym można zadawać pytania dotyczące wystawionych na aukcji przedmiotów.

II faza Okres, w którym klient zapisują się na aukcję i licytuje przedmioty.

Czas odpowiedzi Czas, po którym brak odpowiedzi na komunikat jest interpretowany w pewien sposób (zależny od komunikatu). Określany przez stałą QUERY_TIMEOUT.

3 Cele protokołu

Protokół ma służyć do obsługi aukcji internetowych, bez wykorzystania centralnego serwera dla wszystkich aukcji.

4 Założenia

Protokół działa w warstwie aplikacji. W warstwie transportowej wykorzystuje protokół TCP - potrzebna jest jego niezawodność oraz, głównie ze względu na transfer plików, gwarancja zachowania kolejności wysłanych danych. Aukcja jest jednoznacznie identyfikowana przez adres IP oraz port na którym nasłuchuje serwer. W przypadku udostępnienia tylko jednej aukcji na pod danym adresem IP preferowanym portem jest 1122 (choć nie ma to wpływu na działanie protokołu).

5 Format komunikatów

- Tekst kodowany jest w ASCII (nowsze wersje protokołu mogą używać innego kodowania w celu obsługi znaków rozszerzonych).
- Za porządek oktetów w liczbach przyjmujemy sieciowy.
- Nigdzie nie występuje niejawne wyrównanie, tzn. reprezentacja binarna struktury jest zawsze sumą reprezentacji binarnych jej pól.

Każdy komunikat jest ogólnej postaci:

```
GENERIC_MSG
{
    uint32 Size;
    uint8 Type;
    octet [Size - 5] Message;
}
```

Size Całkowita długość komunikatu, w oktetach (pola Size, Type oraz Message).

Type Typ komunikatu (jedna ze stałych *_MSG).

Message pozostała zawartość komunikatu (zależna od wartości pola Type).

Dla zwięzłości opisu komunikaty utożsamiamy z ich kodami (odpowiadającymi im wartościami pola Type). Wszystkie definicje komunikatów z następnego punktu opisują format pola Message z ogólnej postaci komunikatu, w zależności od pola Type. Jeżeli wielkość tablicy nie jest podana (np. "uint8 [] Tekst"), przyjmujemy że jest maksymalna, tzn. wynosi Size - S oktetów, gdzie S to suma rozmiarów wszystkich innych pól struktury opisującej dany komunikat (włączając w to pola Size oraz Type). W szczególności w strukturze można zdefiniować co najwyżej jedno pole używając tej notacji. Jeżeli Size - S nie jest podzielne przez rozmiar pojedynczego elementu tablicy, przyjmujemy że komunikat jest niepoprawny. Np. definicja VERSION_MSG z kolejnego punktu oznacza, że komunikat VERSION_MSG zawiera pole Size o wartości 4 (Size) + 1 (Type) + 1 (Version), pole Type o wartości VERSION_MSG (interpretowanej jako stała, wykaz stałych znajduje się na końcu dokumentu), oraz pole Version.

6 Opis komunikatów

6.1 Specjalne komunikaty

```
VERSION_MSG
{
    uint8 Version;
}
```

Komunikat wysyłany przez klienta zaraz po nawiązaniu połączenia lub jako odpowiedź serwera na taki sam komunikat wysłany przez klienta.

Version Wersja protokołu, za pomocą której będzie komunikować się klient, lub maksymalna wersja protokołu obsługiwana przez serwer. W przypadku specyfikacji zawartej w tym dokumencie wartość ta powinna wynosić 1.

6.2 Komunikaty fazy I

```
ITEM_MSG
{
    uint8  ItemId;
    uint32 MinPrice;
    uint8  [] Desc;
}
```

Opis przedmiotu wystawionego na aukcji, wysyłany przez serwer.

ItemId Identyfikator przedmiotu, używany w dalszej komunikacji.

MinPrice Cena wywoławcza w PLN.

Desc Tekst opisujący przedmiot.

```
QUERY_MSG
{
    uint8  QueryId;
    uint8  ItemId;
    uint8  [] Text;
}
```

Komunikat wysyłany przez klienta w celu zadania pytania na temat jednego z przedmiotów. W przypadku braku odpowiedzi uznaje się, że sprzedawca zignorował pytanie.

QueryId Identyfikator pytania, wybierany dowolnie przez klienta.

ItemId Identyfikator przedmiotu, którego pytanie dotyczy.

Text Treść pytania, które otrzyma sprzedawca.

Możliwe odpowiedzi: REPLY_MSG, brak (upłynął czas odpowiedzi)

```

REPLY_MSG
{
    uint8 QueryId;
    uint8 TextLength;
    uint8 [TextLength] Text;
    uint8 MimeTypeLength;
    uint8 MimeType[MimeTypeLength]
    octet [] FileData;
}

```

Komunikat wysyłany przez serwer jako odpowiedź na pytanie zawarte w komunikacie QUERY_MSG, którego pole QueryId było równe polu QueryId z tego komunikatu.

QueryId Identyfikator pytania, któremu odpowiada ten komunikat.

TextLength Długość tekstu odpowiedzi.

Text Tekst odpowiedzi.

MimeTypeLength Długość pola MimeType.

MimeType Typ MIME (RFC 2046). Określa format załączonego pliku. Ignorowany w przypadku pola FileData zerowej długości.

FileData Opcjonalna zawartość pliku przesłanego jako załącznik (może być zerowej długości).

```

START2_MSG
{
}

```

Komunikat informujący o przejściu do fazy II - wysyłany przez serwer.

6.3 Komunikaty fazy II

```

ACK_MSG
{
    uint8 Reason;
}

```

Komunikat wysyłany przez serwer w celu potwierdzenia odbioru i przetworzenia innego komunikatu.

Reason Jedna ze stałych RSN_* (patrz 9.2), określająca, czy potwierdzenie jest pozytywne czy negatywne (i co spowodowało ew. błąd).

```

SIGNIN_MSG
{
    uint8 FirstLength;
    uint8 LastLength;
    uint8 [FirstLength] FirstName;
    uint8 [LastLength] LastName;
}

```

Komunikat wysyłany przez klienta w celu dokonania autoryzacji. Dokładne zachowanie w przypadku tego komunikatu opisane jest w 8.3.

FirstLength Długość imienia.

LastLength Długość nazwiska.

FirstName Imię.

LastName Nazwisko.

Możliwe odpowiedzi: ACK_MSG

```

NOTIFY_SIGNIN_MSG
{
    uint8 UserId;
    uint8 FirstLength;
    uint8 LastLength;
    uint8 [FirstLength] FirstName;
    uint8 [LastLength] LastName;
}

```

Komunikat wysyłany przez serwer do wszystkich podłączonych klientów w momencie pomyślnej autoryzacji nowego klienta. Nie jest on wysyłany do klienta który został właśnie autoryzowany.

UserId Identyfikator kupującego, którego autoryzacja spowodowała wysłanie tego komunikatu, używany w dalszej komunikacji.

FirstLength Długość pola FirstName.

LastLength Długość pola LastName.

FirstName Imię nowego kupującego.

LastName Nazwisko nowego kupującego.

```
NOTIFY_BID_MSG
{
    uint8 UserId;
    uint8 ItemId;
    uint32 NewPrice;
}
```

Komunikat wysyłany przez serwer w momencie podbicia ceny przez innego klienta do pozostałych klientów.

UserId Identyfikator kupującego, który podbił cenę.

ItemId Identyfikator przedmiotu, którego cena została podbita.

NewPrice Nowa cena w PLN.

```
BID_MSG
{
    uint8 ItemId;
    uint32 NewPrice;
}
```

Wysyłany przez klienta w celu podbicia ceny przedmiotu.

ItemId Identyfikator przedmiotu.

NewPrice Proponowana cena w PLN.

Możliwe odpowiedzi: ACK_MSG (w wersji 1. protokołu możliwe wartości Reason to RSN_OK, RSN_TOLOW)

```
END2_MSG
{
}
```

Komunikat wysyłany przez serwer, informujący o zakończeniu licytacji.

7 Sytuacje wyjątkowe

- Jeżeli protokół otrzyma niepoprawny komunikat, powinien go zignorować (w szczególności komunikat VERSION_MSG jest niepoprawny podczas trwania fazy I lub II).
- Jeżeli połączenie (z klientem lub serwerem) zostanie zerwane, traktujemy to tak, jakby klient sam się rozłączył (w szczególności nie są anulowane jego wyniki na aukcji).

Zaleca się by implementacja powiadomiła użytkownika o zaistniałej sytuacji wyjątkowej, lub zapisała ten fakt do późniejszego wglądu.

8 Opis komunikacji

Klient, aby wziąć udział w aukcji lub zadać pytanie, musi aktywnie nawiązać połączenie z serwerem udostępniającym daną aukcję. Aukcja identyfikowana jest przez adres IP oraz numer portu udostępniającego ją serwera (każdy serwer obsługuje dokładnie jedną aukcję).

8.1 Negocjacja wersji

Zaraz po nawiązaniu połączenia znajduje się ono w stanie początkowym. Klient zobowiązany jest wysłać komunikat `VERSION_MSG` z polem `Version` ustawionym na numer wersji protokołu, której ma zamiar używać do komunikacji.

Jeżeli dana wersja jest obsługiwana przez serwer, odpowiada on komunikatem `VERSION_MSG` o takiej samej wartości pola `Version`, i dalsza komunikacja odbywa się na zasadach definiowanych przez tę wersję protokołu. W skrajnych przypadkach może to oznaczać zupełnie inny format przyszłych komunikatów lub inny sposób realizacji funkcji protokołu, jednak zalecane jest by nowe wersje protokołu jak najbliżej trzymały się formatu używanego przez starsze wersje.

Jeżeli serwer nie obsługuje żądanej wersji, odpowiada komunikatem `VERSION_MSG` z polem `Version` ustawionym na maksymalną obsługiwaną przez niego wersję protokołu, i połączenie wraca do stanu początkowego (klient powinien zaproponować inną wersję protokołu lub zerwać połączenie).

Komunikacja klienta z serwerem, po negocjacji wersji, podzielona jest na dwie fazy. Po ustaleniu wersji połączenie przechodzi do początku fazy I.

8.2 Faza I

W dowolnym momencie trwania tej fazy klient może otrzymywać od serwera następujące komunikaty:

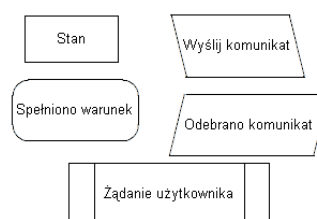
- `ITEM_MSG` - Określający przedmiot który pojawił się na aukcji (w szczególności na początku połączenia klient musi w ten sposób otrzymać listę przedmiotów wystawionych na sprzedaż).
- `REPLY_MSG` - Określający odpowiedź na wcześniej zadane pytanie, udzieloną przez sprzedawcę.
- `START2_MSG` - Wysyłany, kiedy połączenie przechodzi do fazy II, `AUTH_STATE`. Od momentu jego wysłania serwer nie może wysyłać komunikatów fazy I. Od momentu jego odebrania klient nie powinien wysyłać komunikatów fazy I.

W dowolnym momencie trwania tej fazy serwer może otrzymywać od klienta następujące komunikaty:

- `QUERY_MSG` - Określający pytanie zadane sprzedawcy. Jeżeli klient nie uzyska odpowiedzi w czasie `QUERY_TIMEOUT` sekund, lub otrzyma komunikat `START2_MSG`, uznaje się że odpowiedź nigdy nie nadejdzie.

Połączenie może zostać w dowolnym momencie przerwane przez klienta. Jeżeli klient nawiąże połączenie z serwerem po rozpoczęciu aukcji, w tej fazie otrzymuje tylko listę przedmiotów na sprzedaż, zakończoną natychmiastowym komunikatem START2_MSG. Komunikaty fazy I które dotrą do serwera po rozpoczęciu fazy II (po wysłaniu przez serwer komunikatu START2_MSG) są ignorowane.

8.3 Faza II



Oznaczenia używane w rysunkach automatów stanowych:

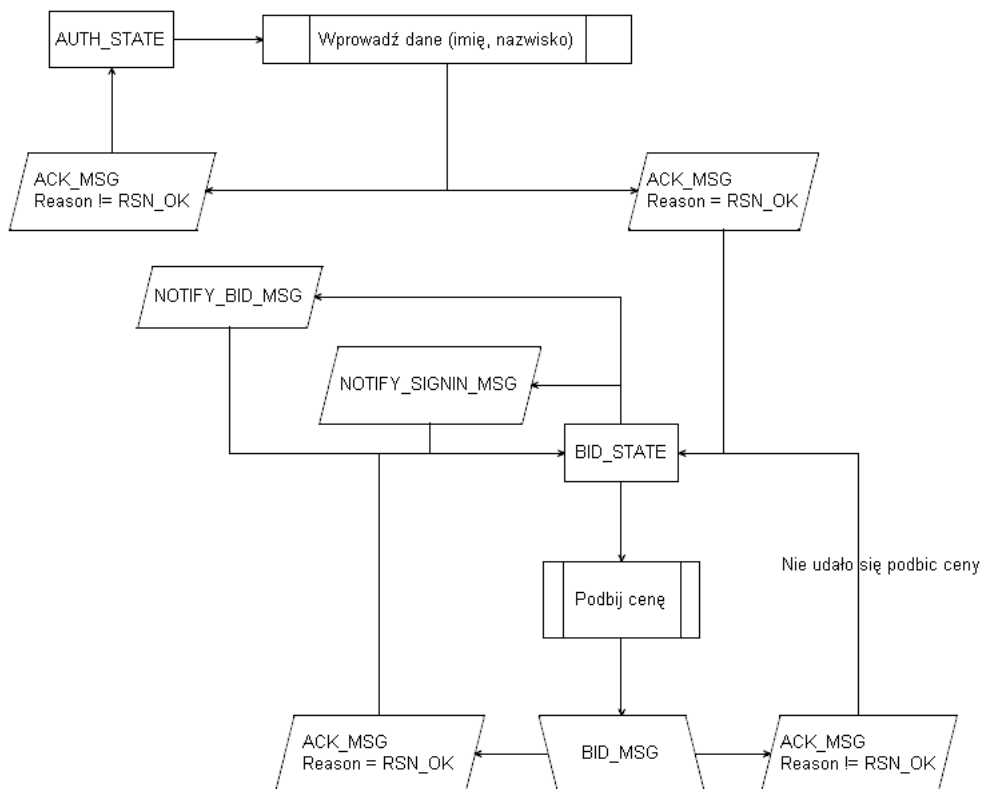
- Klient

AUTH_STATE Początkowy stan tej fazy. Klient powinien wysłać komunikat SIGNIN_MSG w celu uwierzytelnienia. W odpowiedzi otrzymuje komunikat ACK_MSG. Jeżeli autoryzacja powiodła się, pole Reason wynosi RSN_OK i połączenie przechodzi do stanu BID_STATE. W przeciwnym wypadku Reason identyfikuje powstały błąd i klient pozostaje w stanie AUTH_STATE (powtarza próbę autoryzacji).

BID_STATE Stan, w którym klient jest pełnoprawnym uczestnikiem akcji. W tym stanie może otrzymywać komunikaty NOTIFY_SIGNIN_MSG (oznaczające, że do aukcji dołączyła nowa osoba) oraz NOTIFY_BID_MSG (oznaczający, że cena za określony przedmiot wzrosła). Klient może wysłać komunikat BID_MSG w celu licytacji przedmiotu. Jeżeli pomyślnie podbije cenę otrzymuje komunikat ACK_MSG z polem Reason wynoszącym RSN_OK. Jeżeli oferta zostanie odrzucona (np. ktoś w międzyczasie podbił cenę) otrzymuje komunikat ACK_MSG z polem Reason różnym od RSN_OK.

Klient w dowolnym momencie może zerwać połączenie. Nie wyklucza to kupującego z aukcji - może ponownie podłączyć się w celu dalszej licytacji. Traktowany jest wtedy jak nowy klient (jeszcze raz przechodzi przez negocjację wersji i fazę I), za wyjątkiem tego, że w przypadku pomyślnej autoryzacji przy użyciu danych podanych w SIGIN_MSG za pierwszym razem, otrzymuje pierwotny identyfikator użytkownika (nie ma znaczenia, czy klient łączy się z tego samego miejsca w sieci co wcześniej) i utożsamia się z kupującym który wcześniej podał te dane.

Automat stanowy klienta:



- Serwer

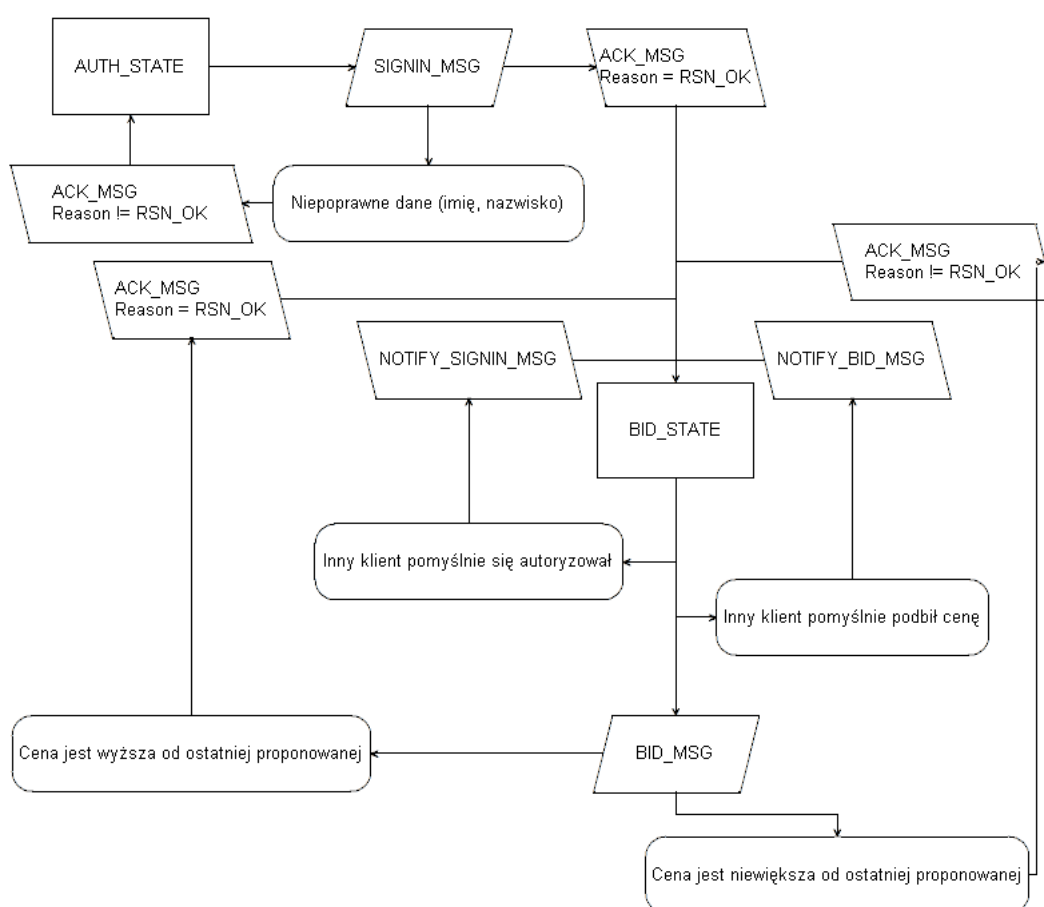
AUTH_STATE Początkowy stan tej fazy. Serwer oczekuje na komunikat SIGNIN_MSG. Jeżeli do serwera nie jest podłączony klient o podanym imieniu i nazwisku, klientowi przydzielany jest unikalny w obrębie aukcji identyfikator (w sposób dowolny, jeżeli dana osoba (jako imię i nazwisko) łączy się po raz pierwszy, lub ostatni używany jeżeli już była podłączona w czasie trwania bieżącej aukcji). Połączenie przechodzi do stanu BID_STATE, a klientowi wysyłany jest komunikat ACK_MSG z polem Reason wynoszącym RSN_OK. W przeciwnym wypadku serwer odpowiada komunikatem ACK_MSG z polem Reason o wartości różnej od RSN_OK.

BID_STATE Stan, w którym klient jest autoryzowany do czynnego udziału w licytacji. Serwer w tym stanie może wysyłać komunikaty NOTIFY_SIGNIN_MSG (z powodów jak opisano powyżej) oraz NOTIFY_BID_MSG. Serwer może otrzymać komunikat BID_MSG - odpowiada wtedy komunikatem ACK_MSG. Jeżeli proponowana cena przedmiotu jest wyższa od aktualnej, pole Reason ustawiane jest na RSN_OK i do pozostałych klientów wysyłany jest odpowiedni komunikat NOTIFY_BID_MSG. W przeciwnym wypadku pole Reason ustawiane jest na wartość różną od RSN_OK. W momencie przejścia do stanu BID_STATE, serwer wysyła do klienta komunikaty NOTIFY_SIGNIN_MSG opisujące już autoryzowanych kupujących, oraz dla

każdego przedmiotu, którego cena została podbita, komunikat NOTIFY_BID_MSG, określający jego nową cenę i kupującego, który ją zaproponował. Do wszystkich pozostałych klientów serwer wysyła pojedynczy komunikat NOTIFY_SIGNIN_MSG, określający kupującego, który właśnie został autoryzowany.

Komunikaty fazy I (QUERY_MSG) w tej fazie są ignorowane, ale nie są uznawane za niepoprawne.

Automat stanowy serwera:



Licytacja kończy się wskutek akcji sprzedającego - połączenie z każdym klientem jest zamykane. Przedmiot uznaje się za sprzedany kupującemu, który jako ostatni pomyślnie podbił jego cenę (informacja o tym nie jest jawnie przesyłana - każdy klient pamięta dla każdego przedmiotu kto dał za niego najwyższą cenę).

9 Wykaz stałych

9.1 Komunikaty

VERSION_MSG = 1
ITEM_MSG = 2
QUERY_MSG = 3
REPLY_MSG = 4
START2_MSG = 5
SIGNIN_MSG = 6
NOTIFY_SIGNIN_MSG = 8
NOTIFY_BID_MSG = 9
ACK_MSG = 10
BID_MSG = 11

9.2 Kody błędów dla ACK_MSG

RSN_OK = 0 (potwierdzenie pozytywne)
RSN_ALREADY = 1 (już jest zapisana osoba o podanym imieniu i nazwisku)
RSN_TOLOW = 2 (proponowana cena jest za mała)

9.3 Inne stałe

QUERY_TIMEOUT = 600 (w sekundach)