

bossaAPI

Podręcznik Użytkownika

bossaNOL3 Wersja 3.1
NOLClientAPI Wersja 10.0.0.72

Spis treści

1. Wprowadzenie.....	4
1.1. Wsparcie techniczne	4
1.2. Wymagania techniczne	4
1.3. Ograniczenia bossaAPI	5
1.4. Dostęp do Rejestru bossaAPI	5
1.5. Wyłączenie/włączenie pozytywnych/negatywnych komunikatów	5
2. Komunikacja FIXML	6
3. Specyfikacja FIXML	8
3.1. Logowanie/Wylogowanie	8
3.2 Stan połączenia	10
3.3 Pobieranie listy papierów	10
3.4 Operacje na filtrze papierów	12
3.5 Status sesji.....	15
3.6 Wyciąg.....	17
3.7 Informacja o opóźnieniach	18
3.8 Wizjer/Outlook.....	18
3.9 Składanie pojedynczego zlecenia.....	19
3.10 Anulowanie zlecenia	22
3.11 Modyfikacja zlecenia.....	23
3.12 Zapytanie o status zlecenia	25
3.13 Wartości atrybutów	26
3.13.1 Typ zlecenia / OrderType	26
3.13.2 Typ daty / TimeInForce	26
3.13.3 Notowania / EntryType	26
3.13.4 Status zlecenia / OrderStatus.....	26
4. ID Komunikatów	27
5. Funkcje biblioteki nolclientapi.....	27
5.1. Funkcja Inicjalizująca bibliotekę	27
5.1.1 Inicjalizacja funkcji zwrotnej statusu aplikacji NOL3.....	27
5.1.2 Funkcja inicjalizująca wskaźnik do listy papierów.....	28
5.1.3 Funkcja importująca listę papierów	28
5.1.4 Zwolnienie struktury danych.....	29

5.1.5	Funkcje modyfikujące filtr papierów w aplikacji zewnętrznej lub NOL3	29
5.1.6	Usunięcie papierów z filtra.....	30
5.1.7	Zapytanie o stan sesji	30
5.1.8	Inicjalizacja funkcji zwrotnej dotyczącej danych online.....	30
5.1.9	Inicjalizacja funkcji zwrotnej dotyczącej statusu zleceń	32
5.1.10	Inicjalizacja funkcji zwrotnej dotyczącej wyciągu	33
5.1.11	Inicjalizacja funkcji zwrotnej dotyczącej wizjera.....	34
5.1.12	Inicjalizacja funkcji zwrotnej dotyczącej opóźnień	34
5.1.13	Funkcja umożliwiająca złożenie, modyfikację, anulację zlecenia oraz otrzymanie informacji o statusie zlecenia.....	35
5.1.14	Odczyt opisu kodu błędu.....	36
5.1.15	Odczyt wersji biblioteki	36
5.1.16	Zakończenie pracy z biblioteką	36
5.2	Przykład działania biblioteki.....	37
5.3	Deskryptor błędów – biblioteka.....	38
5.3.1	INITIALIZE	38
5.3.2	ADDTOFILTER	39
5.3.3	REMFROMFILTER.....	40
5.3.4	CLEARFILTER	41
5.3.5	SHUTDOWN.....	41
5.3.6	SETCALLBACK, SETCALLBACKACCOUNT, SETCALLBACKORDER, SETCALLBACKOUTLOOK, SETCALLBACKDELAY, SETCALLBACKSTATUS	42
5.3.7	APIORDERREQUEST	42
5.3.8	Błędy zwracane w NolRecentInfo	43
5.3.9	SetTradingSess	43
5.3.10	ReleaseTickersList	43
5.3.11	GetTickers.....	44

1. Wprowadzenie

Protokół Financial Information eXchange (FIX) jest otwartym standardem komunikatów stworzonym w celu ułatwienia elektronicznej wymiany danych dotyczących handlu papierami wartościowymi.

W protokole FIXML, który bazuje na FIX, dane prezentowane są w sposób strukturalizowany (XML), a dzięki czemu, uproszczone jest ich przechowywanie, przetwarzanie i przesyłanie.

Poniżej przedstawiamy dokumentację interfejsu programowania aplikacji bossaAPI, który bazuje na protokole FIXML, zapewniając dostęp zdefiniowanych funkcji, za pomocą których możliwe jest pisanie od podstaw własnych programów do zarządzania rachunkiem inwestycyjnym w bossa.pl.

1.1. Wsparcie techniczne

W przypadku pytań lub problemów z aplikacją bossaNOL3 prosimy o kontakt telefoniczny pod numerem **0801 104 104** lub mailowy na adres makler@bossa.pl.

1.2. Wymagania techniczne

bossaAPI jest częścią składową **aplikacji bossaNOL3**, która przeznaczona jest wyłącznie dla następujących systemów operacyjnych:

- Windows 10 (Home/Pro).

Wsparcie techniczne świadczone jest przez DM BOŚ **wyłącznie dla ww. systemów operacyjnych**. W przypadku innych lub pozostałych wersji systemów Windows, aplikacja bossaNOL3 może się nie zainstalować bądź działać niezgodnie z niniejszym Podręcznikiem Użytkownika.

Wygasanie wsparcia dla różnych wersji Windows 10 Home/Pro będzie przebiegało zgodnie z cyklem życia określonym na stronach firmy Microsoft:

<https://support.microsoft.com/pl-pl/help/13853/windows-lifecycle-fact-sheet>

Aplikacja bossaNOL3 dostosowana jest do następujących przeglądarek:

- Google Chrome,
- Mozilla Firefox,
- Microsoft Edge Chromium.

Ważne! W przypadku zmian wprowadzanych przez dostawców przeglądarek powyższa lista może ulec zmianie.

Więcej informacji o aplikacji bossaNOL3 znajduje się w podręczniku użytkownika dostępnym w serwisie transakcyjnym w zakładce **[Notowania]** lub w serwisie informacyjnym <http://bossa.pl/narzedzia/notowania/>

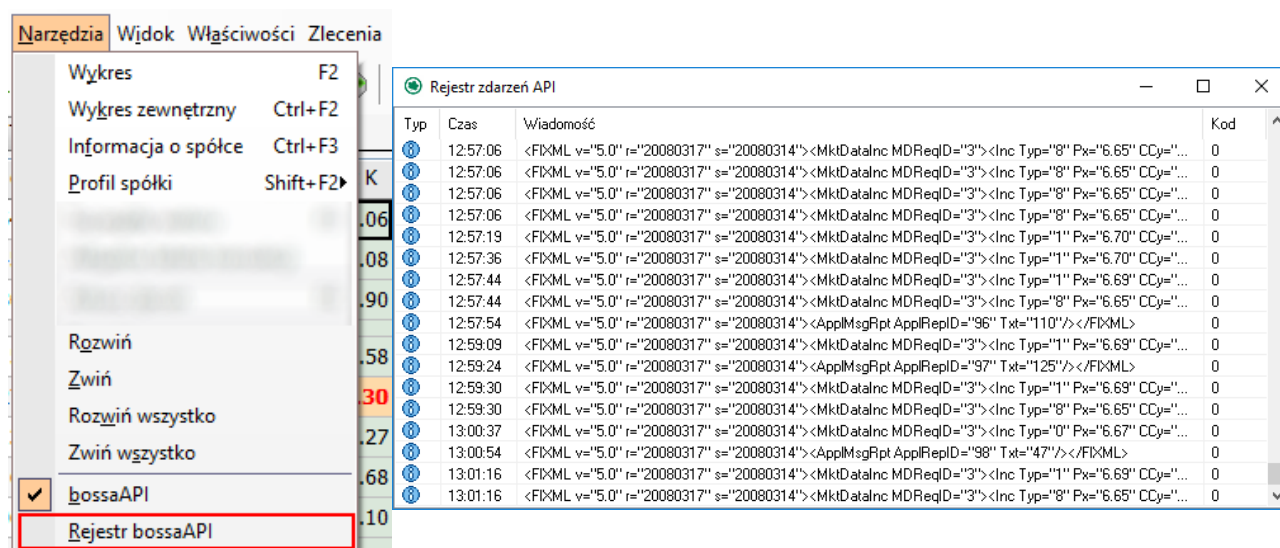
1.3. Ograniczenia bossaAPI

- Maksymalna ilość walorów w filtrze: 100
- Maksymalna ilość zleceń na interwał czasowy wyrażony w milisekundach(10000): 12
- Dzienny limit operacji (zlecenia kupna/sprzedaży, anulaty, modyfikacje) na jednym rachunku: 2000.

Ważne! Interfejs programowania aplikacji bossaAPI nie wspiera obsługi widełek statycznych. Przekroczenie dolnych/górnych widełek statycznych raportowane jest stosownym komunikatem w logach: **Price exceeded static collar**

1.4. Dostęp do Rejestru bossaAPI

Podgląd rejestru bossaAPI jest możliwy w menu **[Narzędzia] -> [Rejestr bossaAPI]**.



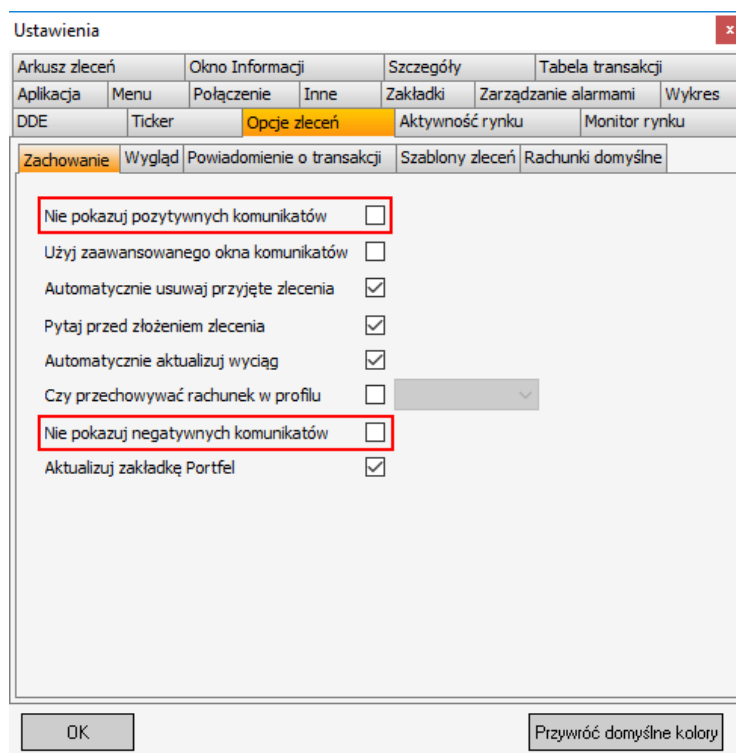
1.5. Wyłączenie/włączenie pozytywnych/negatywnych komunikatów

W przypadku prawidłowego bądź nieprawidłowego złożenia zlecenia w aplikacji bossaNOL3 pojawia się okno Powiadomienia/Komunikatu.

Możliwe jest wyłączenie/włączenie ww. komunikatów w menu **[Właściwości] -> [Opcje zleceń] -> [Zachowanie]**:

- **Nie pokazuj pozytywnych komunikatów** – nie będą prezentowane informacje na temat np. poprawnie przekazanych zleceń/dyspozycji.
- **Nie pokazuj negatywnych komunikatów** – w przypadku włączenia nie będą pokazywane żadne negatywne komunikat, np. związane z odrzuceniem zlecenia.

Zaznaczenie tych opcji ułatwia pracę w aplikacji klienckiej z bossaAPI, gdyż sterowanie nie jest przekazywane do aplikacji bossaNOL3.



2. Komunikacja FIXML

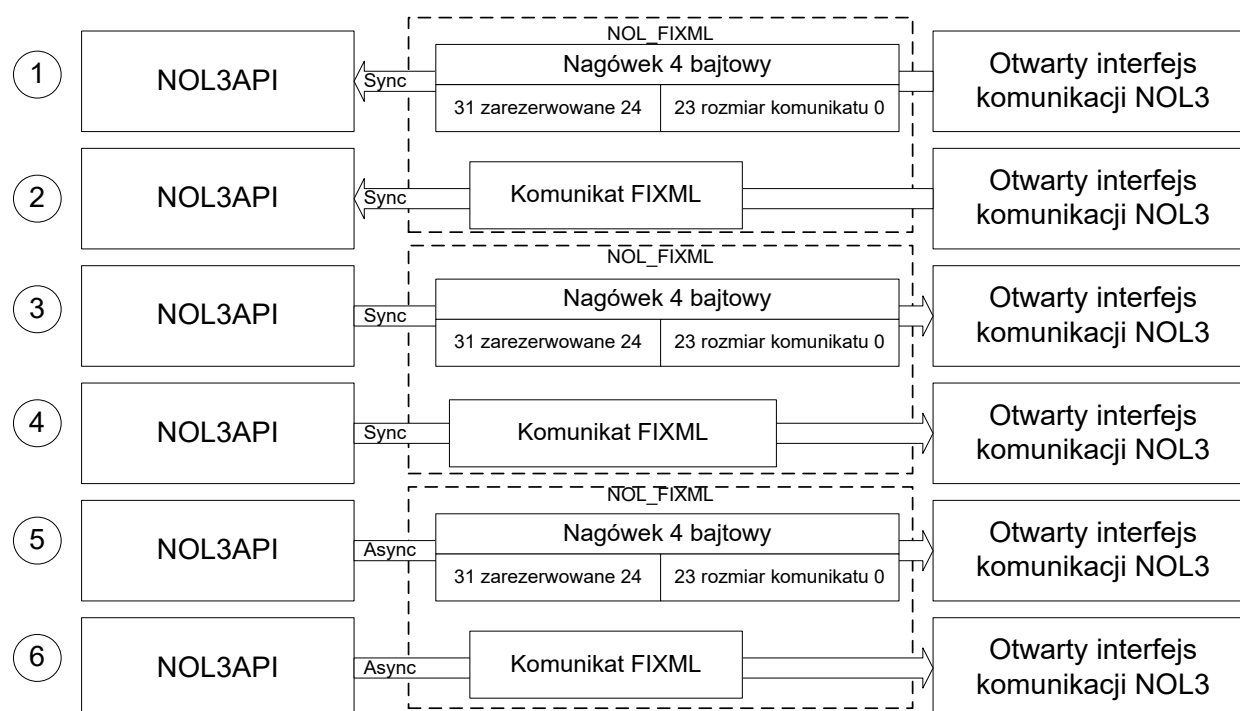
Do komunikacji zostaną użyte dwa sockety:

- **synchroniczny**, w osobnym wątku po stronie otwartego interfejsu komunikacji NOL3,
- **asynchroniczny**, które należy obsłużyć wielowątkowo po stronie otwartego interfejsu komunikacji NOL3.

W celu podłączenia się do API należy skorzystać z następujących rejestrów w HKEY_CURRENT_USER/Software/COMARCH S.A./NOL3/7/Settings:

- **nca_pasync** – wartość portu dla kanału asynchronicznego (wartość domyślna: 24445),
- **nca_psync** – wartość portu dla kanału synchronicznego (wartość domyślna 24444),
- **ncaset_pasync** – flaga informująca czy wartość w nca_pasync jest aktywna (1 - aktywna, 0 – nieaktywna),
- **ncaset_psync** - flaga informująca czy wartość w nca_psync jest aktywna (1 - aktywna, 0 – nieaktywna).

Po uruchomieniu aplikacji NOL3 port dla kanału synchronicznego jest ustawiony, natomiast poprawne zalogowanie się powoduje ustawienie portu dla kanału asynchronicznego.



- 1) Po nawiązaniu komunikacji do NOL3API zostaje wysłany komunikat 4 bajtowy, w którym 24 młodsze bity reprezentują długość komunikatu FIXML, a 8 starszych jest zarezerwowane. Dzięki temu NOL3API ma informację kiedy przestawić socket z trybu odbierania na tryb wysyłania.
- 2) Po utworzeniu bufora o odpowiedniej długości wysłany jest komunikat na podstawie protokołu FIXML. Po wysłaniu komunikatu otwarty interfejs komunikacji NOL3 nasłuchuje odpowiedzi.
- 3) W odpowiedzi NOL3API wysyła komunikat o ustalonej długości zawierający informację na temat długości komunikatu FIXML. Odpowiedź ta pochodzi od aplikacji NOL3 lub DMBOŚ, które mogą być wysłane natychmiast, tak by nie zawieszać działania aplikacji, a odpowiedzi od GPW będą napływać przez socket asynchroniczny.
- 4) Po utworzeniu bufora o odpowiedniej długości po stronie otwartego interfejsu komunikacji NOL3 zostaje wysłany komunikat w oparciu o FIXML. W przypadku większych komunikatów biblioteka winsock2 dzieli komunikaty na paczki (np. przy odbieraniu listy instrumentów), więc należy odbierać dane dopóki są one w buforze. Po otrzymaniu komunikatu sterowanie jest przekazywane dalej.
- 5) Przed wysłaniem komunikatu kanałem asynchronicznym zostanie wysłany identyczny komunikat o długości 4 bajtów jak przy komunikacji synchronicznej
- 6) Po utworzeniu bufora o odpowiedniej długości zostaje wysłany komunikat w oparciu o protokół FIXML.

3. Specyfikacja FIXML

3.1. Logowanie/Wylogowanie

Do logowania zostanie użyty komunikat „UserRequest”. Odpowiedź negatywna lub pozytywna związana z logowaniem zostanie przekazana komunikatem „UserResponse”. Komunikacja odbywać się będzie kanałem synchronicznym. W przypadku błędnego xml, nieznanego komunikatu itp. itd. zostanie wysłany komunikat „BusinessMessageReject”. Komunikat ten będzie używany jako odpowiedź negatywna przy każdym rodzaju komunikacji.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	UserRequest	Synchroniczna	Komunikat logowania
NOL3API→	UserResponse	Synchroniczna	Odpowiedź na komunikat logowania
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Atrybuty pola FIXML			
Atrybut	Wartość	Opis	Wartość
v	String	Wersja FIXML'a	„5.0”
r	String	Dzień wydania zastosowanej wersji FIXML'a	“20080317”
s	String	Dzień wydania zastosowanego FIXML schema	„20080314”

Komunikat FIXML UserRequest			
Atrybut	Wartość	Opis	Wartość
UserRequestID (UserReqID)	String	ID komunikatu	
UserRequestType(UserReqTyp)	int	Rodzaj żądania	1 - logowanie 2 - wylogowanie 4 - status użytkownika
Username	String	Użytkownik	
Password	String	Hasło	

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <UserReq UserReqID="0" UserReqTyp="1" Username="BOS" Password="BOS"/>
</FIXML>
```


Komunikat FIXML UserResponse			
Atrybut	Wartość	Opis	Wartość
UserRequestID(UserReqID)	String	ID żądania użytkownika	
Username	String	Użytkownik lub ID	
UserStatus(UserStat)	int		1 - zalogowanie użytkownika 2 - wylogowanie użytkownika 3 - użytkownik nie istnieje 4 - niepoprawne hasło 5 - inwestor offline 6 - inne 7 - nol offline
UserStatusText(UserStatText)	String		
MktDepth	int	Ilość ofert	0 - 5 ofert (rezerwa na cały arkusz) 1 - 1 oferta 5 - 5 ofert

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <UserRsp UserReqID="0" Username="BOS" MktDepth="5" UserStat="1"/>
</FIXML>
```

Komunikat FIXML BusinessMessageReject			
Atrybut	Wartość	Opis	Wartość
RefMsgType(RefMsgTyp)	String	Komunikat FIXML	BE - logowanie/wylogowanie D - nowe zlecenie F - anulata zlecenia G - modyfikacja zlecenia H - status zlecenia V - notowania online g - status oraz faza sesji
BusinessRejectReason(BizRejRsn)	String	Powód odrzucenia	0 - inny 1 - nieznane ID 2 - nieznany instrument 3 - nieznany typ komunikatu 4 - brak dostępu do aplikacji 5 - błąd xml (brak atrybutu) 6 - brak autoryzacji 7 - brak komunikacji
Text(Txt)	String		

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <BizMsgRej RefMsgTyp="BE" BizRejRsn="5"/>
</FIXML>
```

3.2 Stan połączenia

Stan połączenia jest przekazywany kanałem asynchronicznym za pomocą komunikatu UserResponse.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API→	UserResponse	Asynchroniczna	Odpowiedź na komunikat logowania

Komunikat FIXML UserResponse			
Atrybut	Wartość	Opis	Wartość
UserRequestID(UserReqID)	String	ID żądania użytkownika	
UserStatus(UserStat)	int		6 - inne
UserStatusText(UserStatText)	String		1 - zamknięcie aplikacji NOL3 2 - NOL3 jest offline 3 - NOL3 jest online 4 - aplikacja NOL3 nie jest uruchomiona 5 - Inwestor jest offline 6 - łączenie 7 - łączenie (Inwestor jest offline)

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <UserRsp UserReqID="0" UserStat="6" UserStatText="1"/>
</FIXML>
```

3.3 Pobieranie listy papierów

Z otwartego interfejsu komunikacji NOL3 możemy wysłać komunikat „SecurityListRequest” dzięki któremu mamy możliwość pobrania listy instrumentów w zależności od parametrów „ListReqTyp”, „MktID” oraz „Instrmt”. W odpowiedzi na zapytanie synchroniczne zostanie wysłany komunikat „Security List” w którym znajdować się będą informacje o liście papierów GPW.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	SecurityListRequest	Synchroniczna	Zapytanie - żądanie pobrania listy instrumentów
NOL3API→	SecurityList	Synchroniczna	Odpowiedź, lista instrumentów
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML SecurityListRequest			
Atrybut	Typ	Opis	Wartość
SecurityReqID(ReqID)	String	ID	
SecurityListRequestType(ListReqTyp)	int		0 - jeden instrument (po ISIN lub nazwie) 1 - lista jednego typu instrumentu 4 - cała lista 5 - lista dla jednego kodu rynku
MarketID(MktID)	String	Kod rynku	NM - rynek kasowy DN - rynek pochodny

Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru/skrót	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4
Instrument(Instrmt)/CFIcode(CFI)	String	Typ instrumentu	FFXXXX - futures OCAXXX - opcja amerykańska call OPAXXX - opcja amerykańska put OCEXXX - opcja europejska call OPEXXX - opcja europejska put ESXXXX - akcje, warranty, prawa pierwszeństwa, certyfikaty inwestycyjne i indeksy DBXXXX - obligacje i listy zastawne

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <SecListReq ReqID="0" ListReqTyp="0">
    <Instrmt Sym="COMARCH"/>
  </SecListReq>
</FIXML>
```

Komunikat FIXML SecurityList			
Atrybut	Typ	Opis	Wartość
SecurityReportID(RptID)	Int	ID	
SecurityReqID(ReqID)	string	ID żądania	
MarketID(MktID)	String	Kod rynku	NM - rynek kasowy DN - rynek pochodny
TotNoRelatedSym(TotNoReltdSym)	int	Ilość instrumentów	
SecurityRequestResult(ReqRstl)	int		0 - poprawna odpowiedź 1 - brak papieru 4 - brak listy instrumentów 5 - brak papierów danego typu
SecL/Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru/skrót	np. „COMARCH”
SecL /Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
SecL /(Instrmt)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4
SecL/Instrument(Instrmt)/CFIcode(CFI)	String	Typ instrumentu	FFXXXX - futures OCAXXX - opcja amerykańska call OPAXXX - opcja amerykańska put OCEXXX - opcja europejska call OPEXXX - opcja europejska put ESXXXX - akcje, warranty, prawa pierwszeństwa, certyfikaty inwestycyjne i indeksy DBXXXX - obligacje i listy zastawne
SecL/ Instrmt/SecurityGroup (SecGrp)	String	Grupa instrumentów	

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <SecList RptID="0" ReqID="0" MktID="NM" ReqRslt="0" TotNoReltdSym="1">
    <SecL>
      <Instrmt Sym="COMARCH" Src="4" ID="PLCOMAR00012" SecGrp=" C7" CFI="ESXXXX"/>
    </SecL>
  </SecList>
</FIXML>
```

3.4 Operacje na filtrze papierów

Z otwartego interfejsu komunikacji NOL3 możemy wysłać komunikat „MarketDataRequest” dzięki któremu mamy możliwość dodania walorów do filtra lub wyczyścić filtr znajdujący się po stronie NOL3 (w zależności od pola „SubscriptionRequestType”).

Na podstawie papierów znajdujących się w filtrze będą przychodzić asynchronicznie komunikaty odnośnie danych online.

W odpowiedzi na zapytanie synchroniczne zostanie wysłany „MarketDataSnapshotFullRefresh” z ID odpowiadającemu żądaniu. Komunikat „MarketDataIncrementalRefresh” przychodzi asynchronicznie po zmianie danych na GPW lub zmianie w filtrze.

W celu zmian filtra należy najpierw wysłać komunikat o wyczyszczeniu filtra, a następnie wysłać żądanie z walorami.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	MarketDataRequest	Synchroniczna	Zapytanie - dodanie papierów do filtra lub wyczyszczenie filtra
NOL3API→	MarketDataSnapshotFullRefresh	Synchroniczna	Odpowiedź na MarketDataRequest
NOL3API→	MarketDataRequestReject	Synchroniczna	Odpowiedź z informacją o powodzie odrzucenia zapytania
NOL3API→	MarketDataIncrementalRefresh	Asynchroniczna	Odpowiedź zawierająca zmiany w filtrze lub na GPW
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML MarketDataRequest			
Atrybut	Typ	Opis	Wartość
MDReqID(ReqID)	String	ID	
SubscriptionRequestType(SubReqTyp)	char		1 - żądanie otrzymania image/komunikatów online dla papierów 2 - wyczyszczenie filtra
MarketDepth(MktDepth)	int	Liczba ofert	0 - 5 najlepszych ofert(rezerwa na wszystkie oferty) 1 - najlepsza oferta 2 - 5 najlepszych ofert

MDReqGrp(req)/MDEntryType(Typ)	char	Notowania	pkt 3.13.3
InstrmtMDReqGrp(InstReq)/Symbol(Sym)	String	Nazwa papieru/skrót	np. „COMARCH”
InstrmtMDReqGrp(InstReq)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
InstrmtMDReqGrp(InstReq) /SecurityIDSource(Src)	int	rodzaj kodu dla ID	4

Przykład:

Zapytanie o notowania online dla waloru PGNIG, LOTOS.

```
<FIXML v="5.0" r="20080317" s="20080314">
  <MktDataReq ReqID="1" SubReqTyp="1" MktDepth="0">
    <req Typ="0"/><req Typ="1"/><req Typ="2"/><req Typ="B"/>
    <req Typ="C"/><req Typ="3"/><req Typ="4"/><req Typ="5"/>
    <req Typ="7"/><req Typ="r"/><req Typ="8"/>
    <InstReq>
      <Instrmt ID="PLPGNIG00014" Src="4"/>
      <Instrmt ID="PLLOTOS00025" Src="4"/>
    </InstReq>
  </MktDataReq>
</FIXML>
```

Przykład:

Wyczyszczenie filtra:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <MktDataReq ReqID="0" SubReqTyp="2">
  </MktDataReq>
</FIXML>
```

Komunikat FIXML MarketDataRequestReject			
Atrybut	Typ	Opis	Wartość
MDReqID(ReqID)	String	ID MarketDataRequest	
MDReqRejReason(ReqRejResn)	char	Powód odmowy	0 - Nieznany walor 1 - Duplikat MDReqID 4 - Błąd w polu SubscriptionRequestType 5 - Niewspierana liczba ofert
Text(Txt)	String		

Przykład (Sync) - Odpowiedź w przypadku zapytania o walor, który nie istnieje:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <MktDataReqRej ReqID="0" ReqRejReason="0"/>
</FIXML>
```

Komunikat FIXML MarketDataIncrementalRefresh			
Atrybut	Typ	Opis	Wartość
MDReqID(ReqID)	String	ID MarketDataRequest	
MDIncGrp(Inc)/MDEntryType(Typ)	char	Notowania	pkt 3.13.3
MDIncGrp(Inc)/Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
MDIncGrp(Inc)/Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
MDIncGrp(Inc)/Instrument(Instrmt)/SecurityIDSource(Src)	int	Rodzaj kodu pola ID	4
MDIncGrp(Inc)/MDEntryPx(Px)	float	Cena	
MDIncGrp(Inc)/MDPriceLevel(MDPxLvl)	Int	Pozycja oferty w arkuszu	1 - 5
MDIncGro(Inc)/Currency(Ccy)	float	Waluta	PLN
MDIncGrp(Inc)/MDEntrySize(Sz)	float	Rozmiar	
MDIncGrp(Inc)/NumberOfOrders(NumOfOrds)	int	Liczba ofert	
MDIncGrp(Inc)/MDEntryDate(Dt)	String	Data notowania	YYYYMMDD
MDIncGrp(Inc)/MDEntryTime(Tm)	String	Czas notowania	HH:MM:SS
MDIncGrp(Inc)/TurnoverValue(Tov)	float	Wartość obrotu	

Przykład (Async):

```

<FIXML v="5.0" r="20080317" s="20080314">
  <MktDataInc MDReqID="1">
    <Inc Typ="1" Px="6.74" CCy="PLN" Sz="7199" NumOfOrds="3" MDPxLvl="5">
      <Instrmt Sym="PGNIG" ID="PLPGNIG00014" Src="4"/>
    </Inc>
    <Inc Typ="1" Px="6.71" CCy="PLN" Sz="82846" NumOfOrds="7" MDPxLvl="2">
      <Instrmt Sym="PGNIG" ID="PLPGNIG00014" Src="4"/>
    </Inc>
  </MktDataInc>
</FIXML>

```

	Px	Ccy	sz	NumOfOrds	MDPxLvl	Dt	Tm	Tov
Bid Oferta kupna Offer Oferta sprzedaży	Cena oferty	waluta	Wolumen oferty	Liczba ofert	Pozycja ofert w arkuszu	-	-	-
Trade/ ostatnia transakcja	Cena transakcji	waluta	Wolumen transakcji	-	-	Data	czas	-
Trade Volume/wolumen obrotu	-	-	Wolumen obrotu	-	-	-	-	Wartość obrotu
Open Interest LOP	-	-	LOP	-	-	-	-	-

Index Value	Wartość indeksu	-	-	-	-	-	-	Wartość obrotu na indeksie
Opening Price kurs otwarcia closing price kurs zamknięcia	Kurs otwarcia kurs zamknięcia	waluta	-	-	-	-	-	Wartość obrotu otwarcia/zamknięcia
Trading Session High Price Kurs maksymalny Trading Session Low Price kurs minimalny	kurs maksymalny kurs minimalny	waluta	-	-	-	-	-	-
Reference price	Kurs odniesienia	waluta	-	-	-	-	-	-

W przypadku zrealizowania lub oferty kupna lub sprzedaży, przy braku innych ofert jest wysyłana oferta z pustymi wartościami.

Komunikat FIXML MarketDataSnapshotFullRefresh			
Atrybut	Typ	Opis	Wartość
MDReqID(ReqID)	String	ID żądania wyczyszczenia filtra lub dodania papierów	

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <MktDataFull>
    <ReqID ="1"/>
  </MktDataFull>
</FIXML>
```

3.5 Status sesji

Z otwartego interfejsu zapytanie o status sesji odbywa się za pomocą „TradingSessionStatusRequest”. Za pomocą pola „SubscriptionRequestType” ustawiamy możliwość otrzymywania komunikatów kanałem asynchronicznym. W odpowiedzi na „TradingSessionStatusRequest” zostanie wysłany w przypadku pozytywnej odpowiedzi, pusty komunikat, w przypadku negatywnej, z informacją o powodzie odrzucenia. Informację, które będą napływać kanałem asynchronicznym, dotyczą papierów w filtrze.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	TradingSessionStatusRequest	Synchroniczna	Zapytanie
NOL3API→	TradingSessionStatus	Synchroniczna	Odpowiedź
NOL3API→	TradingSessionStatus	Asynchroniczna	Odpowiedź
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML TradingSessionStatusRequest			
Atrybut	Typ	Opis	Wartość
TradSesReqID(ReqID)	String	ID	

SubscriptionRequestType (SubReqTyp)	char	Dystrybucja/Subskrybcja	1 - komunikaty online 2 - anulowanie komunikatów online
--	------	-------------------------	--

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <TrdgSesStatReq ReqID="0" SubReqTyp="1"/>
</FIXML>
```

Komunikat FIXML TradingSessionStatus			
Atrybut	Typ	Opis	Wartość
TradSesReqID(ReqID)	String	ID	
TradingSessionID(SesID)	String	Identyfikator sesji	1 - dzień
TradingSessionSubID(SesSub)	String	Faza podczas sesji	C - Konsultacja nadzoru P - Przed otwarciem E - Interwencja O - Otwarcie R - Dogrywka S - Sesja notowań ciągłych N - Interwencja nadzoru F - Konsultacja nadzoru B - Po sesji
TradSesStatus(Stat)	String	Status sesji	AR – Równoważenie AS - Zakaz obrotu, zlecenia tak A - Papier w obrocie IR - Bez dogrywki - zlecenia nie IS - Zakaz obrotu AG - Zamrożenie instrumentu I - Zakaz obrotu
TradSesStatusRejReason(StatRejRsn)	int	Powód odrzucenia żądania	1 - błędny ID sesji 99 - inny
Instrument(Instrm)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrm)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrm)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <TrdgSesStat ReqID="1" SesSub="S">
    <Instrmt Sym="PGNIG" ID="PLPGNIG00014" Src="4" />
  </TrdgSesStat>
</FIXML>
```


3.6 Wyciąg

W standardzie FIXML nie istnieje odpowiedni komunikat obsługujący informację na temat wyciągu. Komunikaty przychodzą kanałem asynchronicznym w komunikacie „XML_non_FIX”. Użytkownik nie będzie miał możliwości wysłania żądania na temat wyciągu.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API→	XML_non_FIX	Asynchroniczna	Komunikat odnośnie wyciągu

Komunikat XML_non_FIX Statement			
Atrybut	Typ	Opis	Wartość
Statement/Account(Acct)	String	Numer konta	
Statement/type	char	Typ rachunku	M- kasowy, P – derywaty
Statement/ike	char	Rachunek IKE	T – tak, N - nie
Statement/Acct/Fund/name	String	Nazwa szczegółu na rachunku	
Statement/Acct/Fund/value	String	Wartość szczegółu na rachunku	
Statement/Acct/Position/Acc110	String	Ilość posiadanych aktywów	
Statement/Acct/Position/Acc120	String	Ilość aktywów zablokowanych	
Statement/Acct/Position/Sym	String	Nazwa papieru	np. „COMARCH”
Statement/Acct/Position/ID	String	ISIN	np. „PLCOMAR00012”
Statement/Acct/Position/Src	int	rodzaj kodu dla ID	4

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314"><Statement Acct="00-22-000000" type="P"
ike="N" ><Fund name="SecValueSum" value="0.00"/><Fund name="Cash" value="1.00"
/><Fund name="Deposit" value="0.00" /><Fund name="CashBlocked" value="0.00"
/><Fund name="BlockedDeposit" value="0.00" /><Fund name="SecSafeties"
value="0.00" /><Fund name="FreeDeposit" value="0.00" /><Fund
name="SecSafetiesUsed" value="0.00" /><Fund name="OptionBonus" value="0.00"
/><Fund name="PortfolioValue" value="1.00" /></Statement><Statement Acct="00-55-
000000" type="M" ike="N" ><Position Acc110="1" Acc120="0"><Instrmt
ID="PLPGER000010" Src="4" Sym="PGE"/></Position><Position Acc110="2"
Acc120="0"><Instrmt ID="PLPGNIG00014" Src="4" Sym="PGNIG"/></Position><Position
Acc110="1" Acc120="0"><Instrmt ID="PLTAURN00011" Src="4"
Sym="TAURONPE"/></Position><Fund name="SecValueSum" value="29.84"/><Fund
name="Cash" value="8.10" /><Fund name="Liabilities" value="0.00" /><Fund
name="Recivables" value="25.46" /><Fund name="MaxBuy" value="111.87" /><Fund
name="CashRecivables" value="33.56" /><Fund name="MaxOtpBuy" value="111.87"
/><Fund name="CashBlocked" value="0.00" /><Fund name="LiabilitiesLimitMax"
value="0.00" /><Fund name="RecivablesBlocked" value="0.00" /><Fund
name="PortfolioValue" value="63.40" /></Statement><Statement Acct="076490"
type="P" ike="N" ><Fund name="SecValueSum" value="0.00"/><Fund name="Cash"
value="0.00" /><Fund name="Deposit" value="0.00" /><Fund name="CashBlocked"
value="0.00" /><Fund name="BlockedDeposit" value="0.00" /><Fund
name="FreeDeposit" value="0.00" /><Fund name="PortfolioValue" value="0.00"
/></Statement><Statement Acct="600988" type="M" ike="Z" ><Fund name="SecValueSum"
value="0.00"/><Fund name="Cash" value="0.00" /><Fund name="Liabilities"
value="0.00" /><Fund name="Recivables" value="0.00" /><Fund name="MaxBuy"
```

```
value="0.00" /><Fund name="CashRecivables" value="0.00" /><Fund name="MaxOtpBuy"
value="0.00" /><Fund name="CashBlocked" value="0.00" /><Fund
name="LiabilitiesLimitMax" value="0.00" /><Fund name="RecivablesBlocked"
value="0.00" /><Fund name="PortfolioValue" value="0.00" /></Statement><Statement
Acct="800594" type="M" ike="T" ><Fund name="SecValueSum" value="0.00"/><Fund
name="Cash" value="0.00" /><Fund name="Liabilities" value="0.00" /><Fund
name="Recivables" value="0.00" /><Fund name="MaxBuy" value="0.00" /><Fund
name="CashRecivables" value="0.00" /><Fund name="MaxOtpBuy" value="0.00" /><Fund
name="CashBlocked" value="0.00" /><Fund name="LiabilitiesLimitMax" value="0.00"
/><Fund name="RecivablesBlocked" value="0.00" /><Fund name="PortfolioValue"
value="0.00" /></Statement></FIXML>
```

3.7 Informacja o opóźnieniach

Na podstawie mechanizmu zaimplementowanego w NOL3 do otwartego interfejsu komunikacji zostanie wysłany komunikat kanałem asynchronicznym z informacjami na temat opóźnień.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API→	ApplicationMessageReport	Asynchroniczna	Komunikat odnośnie opóźnień

Komunikat XML_non_FIX ApplicationMessageReport			
Atrybut	Typ	Opis	Wartość
ApplReportID(ApplRepID)	String	ID komunikatu	
Text(Txt)	String	String zawierający informacje o opóźnieniu	Wartość jest wyrażona w milisekundach

Przykład (Async):

```
<FIXML v="5.0" r="20080317" s="20080314">
  <AppIMsgRpt AppIRepID="1" Txt="22"/>
</FIXML>
```

3.8 Wizjer/Outlook

Komunikaty dla wizjera przychodzą asynchronicznie komunikatem FIXML „News”.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API→	News	Asynchroniczna	Komunikaty spływające z DMBOŚ

Komunikat FIXML News			
Atrybut	Typ	Opis	Wartość
OrigTime(OrigTm)	String	Czas powstania wiadomości	YYYYMMDD-HH:MM:SS
Headline	String	Nagłówek wiadomości	
<![CDATA[News]]>	String	News = Tekst wiadomości	

Przykład (Async):

```
<FIXML v="5.0" r="20080317" s="20080314">
  <News OrigTm="20080910-10:12:23" Headline
    ="Wiadomosci"><![CDATA[ Wiadomosc]]></News>
</FIXML>
```

3.9 Składanie pojedynczego zlecenia

Aby złożyć pojedyncze zlecenie należy wysłać komunikat FIXML „NewOrderSingle” kanałem synchronicznym. W odpowiedzi synchronicznej NOL potwierdzi odbiór lub odrzucenie komunikatu. Poprzez socket asynchroniczny zostanie przysłana odpowiedź w postaci komunikatu „Execution Report”.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	NewOrderSingle	Synchroniczna	Komunikat – zapytanie odnośnie zlecenia
NOL3API→	ExecutionReport	Synchroniczna	Odpowiedź DMBOS
NOL3API→	ExecutionReport	Asynchroniczna	Odpowiedź GPW
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML NewOrderSingle			
Atrybut	Typ	Opis	Wartość
ClOrdID(ID)	String	ID zlecenia	
TradeDate(TrdDt)	String	Dzień sesji	
Account(Acct)	String	Nr konta	
MinQty	float	Ilość minimalna	
DisplayInstruction(DisplyInstr)/DisplayQty	float	Wartość ujawniona	
Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4
Side	char	Strona	1 - Kupno 2 - Sprzedaż
TransactTime(TxnTm)	String	Czas wygenerowania zlecenia przez klienta	YYYYMMDD-HH:MM:SS
OrderQtyData(OrdQty)/OrderQty(Qty)	float	Ilość	
OrdType(OrdTyp)	char	Typ zlecenia	pkt 3.13.1
Price(Px)	float	Cena	
StopPx(StopPx)	float	Limit aktywacji	
Currency(Ccy)	string	Waluta	PLN
TimeInForce(TmInForce)	char	Typ daty	pkt 3.13.2
ExpireDate(ExpireDt)	String	Data ważności (do)	YYYYMMDD
TriggerringInstruction(TrgrInstr)/TriggerType(TrgrTyp)	char	Zlecenia DDM+ - po cenie	4

TriggerringInstruction(TrgrInstr)/TriggerAction(TrgrActn)	char	Zlecenia DDM+ - uaktywnienie	1
TriggerringInstruction(TrgrInstr)/TriggerPrice(TrgrPx)	float	Zlecenia DDM+ -Cena uaktywnienia	
TriggerringInstruction(TrgrInstr)/TriggerPriceType(TrgrPxTyp)	char	Zlecenia DDM+ - ostatnia transakcja	2
DeferredPaymentType(DefPayTyp)	char	OTP	T/P

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
<Order ID="0" TrdDt="20170802" Acct="00-55-000000" Side="1" TxnTm="20170802-10:16:27" OrdTyp="L" Px="6.530000" Ccy="PLN" TmInForce="0">
    <Instrmt ID="PLPGNIG00014" Src="4"/>
    <OrdQty Qty="1"/>
</Order>
</FIXML>
```

Przykład: Zlecenie PEG E bez limitu dodatkowego:

```
<FIXML v="5.0" r="20080317" s="20080314">
<Order ID="0" TrdDt="20170802" Acct="00-55-000000" Side="1" TxnTm="20170802-10:18:47" OrdTyp="E" Ccy="PLN" TmInForce="0">
    <Instrmt ID="PLPGNIG00014" Src="4"/>
    <OrdQty Qty="1"/>
</Order>
</FIXML>
```

Przykład: Zlecenie PEG G z limitem dodatkowym definiowanym w atrybucie „StopPx”:

```
<FIXML v="5.0" r="20080317" s="20080314">
<Order ID="0" TrdDt="20170802" Acct="00-55-000000" Side="1" TxnTm="20170802-10:21:35" OrdTyp="G" StopPx="6.750000" Ccy="PLN" TmInForce="0">
    <Instrmt ID="PLPGNIG00014" Src="4"/>
    <OrdQty Qty="1"/>
</Order>
</FIXML>
```

Komunikat FIXML ExecutionReport			
Atrybut	Typ	Opis	Wartość
OrderID(OrdID)	String	ID zlecenia nadawany przez DM	
SecondaryOrderID(OrdID2)	String	Numer zlecenia nadawany przez DM	

ClOrdID(ID)	String	ID zlecenia nadawane przez klienta lub bibliotekę kliencką	
OrdStatusReqID(StatReqID)	String	Identyfikator żądania z OrderStausRequest	
ExecID	String	ID wykonania zlecenia	
ExecType(ExecTyp)	char	Typ wykonania	0 - Nowy F - transakcja 4 - Anulowanie E - Modyfikacja 6 - w trakcie anulowania 8 - Odrzucone I - Status zlecenia
OrdStatus(Stat)	char	Status zlecenia	pkt 3.13.4
OrdRejReason(RejRsn)	int	Powód odrzucenia zlecenia	98 – niezgodność ID 99 - inny
Account(Acct)	String	Konto	
Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4
Side	char	Strona	1 - Kupno 2 - Sprzedaż
OrderQtyData(OrdQty) /OrderQty(Qty)	float	Ilość	
OrdType(Typ)	char	Typ zlecenia	pkt 3.13.1
Price(Px)	float	Cena	
StopPx	float	Limit aktywacji	
Currency(Ccy)	String	Waluta	„PLN”
TimeInForce(TmInForce)	char	Typ daty	pkt 3.13.2
ExpireDate(ExpireDt)	String	dla TmInForce = 6	YYYYMMDD
LastPx>LastPx)	float	Cena ostatniej transakcji	
LastQty>LastQty)	float	Ilość zrealizowana podczas ostatniej transakcji	
LeavesQty	float	Ilość pozostała w zleceniu	
CumQty	float	Ilość wypełniona w zleceniu	
TransactTime(TxnTm)	String	Czas transakcji	YYYYMMDD-HH:MM:SS
CommisionData(Comm)/Commision(Comm)	float	Prowizja dla transakcji/zlecenia	

(Comm)/CommType(CommTyp)	char	Typ prowizji	3 - wartość absolutna
NetMoney(NetMny)	float	Wartość netto transakcji/zlecenia	
MinQty	float	Wartość minimalna	
DisplayInstruction(DisplyInstr)/DisplayQty	float	Ilość ujawniona	
Txt	String	Dowolny tekst	Powód odrzucenia
TriggerringInstruction(TrgrlInstr)/TriggerType(TrgrTyp)	char	Zlecenia DDM+ - po cenie	4
TriggerringInstruction(TrgrlInstr)/TriggerAction(TrgrActn)	char	Zlecenia DDM+ - uaktywnienie	1
TriggerringInstruction(TrgrlInstr)/TriggerPrice(TrgrPx)	float	Zlecenia DDM+ -Cena uaktywnienia	
TriggerringInstruction(TrgrlInstr)/TriggerPriceType(TrgrPxTyp)	char	Zlecenia DDM+ - ostatnia transakcja	2
DeferredPaymentType(DefPayTyp)	char	OTP	T/P

Przykład (Sync.):

```
<FIXML v="5.0" r="20080317" s="20080314">
<ExecRpt ID="0" OrdID="530469746" OrdID2="103282208" ExecID="0" ExecTyp="0"
Stat="0" Acct="00-55-000000" Side="1" TmInForce="0" OrdTyp="L" Px="6.57"
TxnTm="20170802-10:24" Ccy="PLN" CumQty="0" LeavesQty="1" MinQty="0"
DefPayTyp="N" >
    <Instrmt Sym="PGNIG" Src="4" ID="PLPGNIG00014"/>
    <DsplyInstr DisplayQty="0"/>
    <OrdQty Qty="1"/>
</ExecRpt>
</FIXML>
```

3.10 Anulowanie zlecenia

W celu anulowania zlecenia należy wysłać komunikat „OrderCanelRequest”. Na poziomie NOL3 zostanie zwrócony kod informujący o poprawności komunikatu. Kanałem asynchronicznym zostaną wysłane komunikaty odnośnie przyjęciu lub odrzuceniu anulowania zlecenia.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	OrderCanelRequest	Synchroniczna	Zapytanie odnośnie anulowania zlecenia
NOL3API→	ExecutionReport	Synchroniczna	Odpowiedź pozytywna DMBOS
NOL3API→	ExecutionReport	Asynchroniczna	Odpowiedź pozytywna GPW
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML OrderCanelRequest			
Atrybut	Typ	Opis	Wartość

CIOrdID(ID)	String	ID anulaty	
OrigCIOrdID(OrigID)	String	ID zlecenia nadawane przez klienta lub bibliotekę kliencką. Ważne przez jedną „sesję API”.	
OrderID(OrdID)	String	ID zlecenia nadawany przez DM	
SecondaryOrderID(OrdID2)	String	Numer zlecenia nadawany przez DM	
Account(Acct)	String	Nr rachunku	
Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(src)	int	rodzaj kodu pola ID	4
Side	char	Strona	1 - Kupno / 2 - Sprzedaż
TransactTime(TxnTm)	String	Czas utworzenia zlecenia	YYYYMMDD-HH:MM:SS
OrderQtyData(OrdQty)/OrderQty(Qty)	float	Ilość	
Text(Txt)	String	Dowolny tekst	

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <OrdCxlReq ID="0" OrdID="530471025" Acct="00-55-000000" Side="1" TxnTm="20170802-10:27:39">
    <Instrmt ID="PLPGNIG00014" Src="4"/>
    <OrdQty Qty="1"/>
  </OrdCxlReq>
</FIXML>
```

3.11 Modyfikacja zlecenia

Komunikacja jest identyczna jak w przypadku anulowania zlecenia.

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	OrderCanelReplaceRequest	Synchroniczna	Zapytanie odnośnie modyfikacji zlecenia
NOL3API→	ExecutionReport	Synchroniczna	Odpowiedź pozytywna DMBOS
NOL3API→	ExecutionReport	Asynchroniczna	Odpowiedź pozytywna GPW
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML OrderCanelReplaceRequest			
Atrybut	Typ	Opis	Wartość
TradeDate(TrdDt)	String	Dzień sesji	YYYYMMDD
CIOrdID(ID)	String	ID zlecenia	
OrigCIOrdID(OrigID)	String	ID zlecenia nadawane przez klienta lub bibliotekę kliencką. Ważne przez jedną „sesję API”.	

OrderID(OrdID)	String	ID zlecenia nadawane przez DM	
SecondaryOrderID(OrdID2)	String	Numer zlecenia nadawany przez DM	
Account(Acct)	String	Nr konta	
MinQty	float	Wartość minimalna	
DisplayInstruction(DsplyInstr)/DisplayQty	float	Ilość ujawniona	
Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(src)	int	rodzaj kodu dla ID	4
Side		Strona	1 - Kupno 2 - Sprzedaż
TransactTime(TxnTm)	String	Czas transakcji	YYYYMMDD-HH:MM:SS
OrderQtyData(OrdQty)/OrderQty(Qty)	float	Ilość	
OrdType(Typ)	char	Typ zlecenia	pkt 3.13.1
Price(Px)	float	Cena	
StopPx	float	Limit aktywacji	
Currency(Ccy)	String	Waluta	„PLN”
TimeInForce(TmInForce)	char	Typ daty	pkt 3.13.2
ExpireDate(ExpireDt)	String	Data ważności	YYYYMMDD
Text(Txt)	String	Dowolny tekst	
TriggerringInstruction(TrgrInstr)/TriggerType(TrgrTyp)	char	Zlecenia DDM+ - po cenie	4
TriggerringInstruction(TrgrInstr)/TriggerAction(TrgrActn)	char	Zlecenia DDM+ - uaktywnienie	1
TriggerringInstruction(TrgrInstr)/TriggerPrice(TrgrPx)	float	Zlecenia DDM+ -Cena uaktywnienia	
TriggerringInstruction(TrgrInstr)/TriggerPriceType(TrgrPxTyp)	char	Zlecenia DDM+ - ostatnia transakcja	2
DeferredPaymentType(DefPayTyp)	char	OTP	T/P

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <OrdCxlRplcReq ID="0" OrdID="530471817" OrdID2="103282541" TrdDt="20170802"
    Acct="00-55-000000" Side="1" TxnTm="20170802-10:29:47" OrdTyp="L" Px="6.590000"
    Ccy="PLN" TmInForce="0">
    <Instrmt ID="PLPGNIG00014" Src="4"/>
    <OrdQty Qty="1"/>
  </OrdCxlRplcReq>
```


</FIXML>

3.12 Zapytanie o status zlecenia

Kierunek	Komunikat FIXML	Typ komunikacji	Opis
NOL3API←	OrderStatusRequest	Synchroniczna	Zapytanie
NOL3API→	ExecutionReport	Synchroniczna	Odpowiedź
NOL3API→	BusinessMessageReject	Synchroniczna	Odpowiedź negatywna

Komunikat FIXML OrderStatusRequest			
Atrybut	Typ	Opis	Wartość
OrderID(OrdID)	String	ID zlecenia nadawane przez DM o którego status się pytamy	
ClOrdID(ID)	String	ID zlecenia nadawany przez klienta lub bibliotekę kliencką	
OrigClOrdID(OrigID)	String	ID zlecenia nadawane przez klienta lub bibliotekę kliencką. Ważne przez jedną „sesję API”.	
OrderStatusReqID(StatReqID)	String	Identyfikator żądania o status	
Account(Acct)	String	Nr konta	
Instrument(Instrmt)/Symbol(Sym)	String	Nazwa papieru	np. „COMARCH”
Instrument(Instrmt)/SecurityID(ID)	String	ISIN	np. „PLCOMAR00012”
Instrument(Instrmt)/SecurityIDSource(Src)	int	rodzaj kodu dla ID	4
Side	char	Strona	1 - Kupno 2 - Sprzedaż

Przykład:

```
<FIXML v="5.0" r="20080317" s="20080314">
  <OrdStatReq StatReqID="5" OrdID="530478088" Acct="00-55-000000" Side="1">
    <Instrmt Sym="PGNIG" Src="4" ID="PLPGNIG00014"/>
  </OrdStatReq>
</FIXML>
```

3.13 Wartości atrybutów

3.13.1 Typ zlecenia / OrderType

- 1 - PKC - po każdej cenie
- L - Limit
- 3 - Stop / Stop Loss
- 4 - Stop Limit
- K PCR - po cenie rynkowej
- K PCRO - przy TimeInForce 2 (na otwarciu) lub 7 (na zamknięciu).
- E - PEG
- G - PEG z limitem

3.13.2 Typ daty / TimeInForce

- 0 - dzień
- 1 - do anulowania
- 2 - przed otwarciem
- 3 - Wykonaj albo anuluj WiA
- 4 - Wykonaj lub anuluj WuA
- 6 - do dnia
- 7 - na zamknięcie
- f - na najbliższy fixing
- t - do czasu

3.13.3 Notowania / EntryType

- 0 - Bid - Oferta kupna
- 1 - Offer - Oferta sprzedaży
- 2 - Trade - ostatnia transakcja
- B - Trade Volume - wolumen obrotu
- C - Open Interest - LOP
- 3 - Index Value - wartość indeksu
- 4 - Opening Price - kurs otwarcia
- 5 - Closing Price - kurs zamknięcia
- 7 - Trading Session High Price - kurs maksymalny
- 8 - Trading Session Low Price - kurs minimalny
- r - Reference price - kurs odniesienia

3.13.4 Status zlecenia / OrderStatus

- 0 - nowe
- C - archiwalne
- E - w trakcie modyfikacji
- 1 - wykonane/aktywne
- 2 - wykonane

4 - anulowane
6 - w trakcie anulaty
8 – odrzucone

4. ID Komunikatów

W przypadku komunikatów:

- UserRequest
- NewOrderSingle
- OrderCancelReplaceRequest
- OrdrCancelRequest
- OrderStatusRequest
- TradingSessionStatusRequest
- MarketDataRequest

należy zadbać aby ID ww. komunikatów były wartościami rosnącymi.

5. Funkcje biblioteki nolclientapi

Biblioteka umożliwia integrację NOL3 z dowolną aplikacją zewnętrzną wykorzystując ogólnodostępne procedury.

5.1. Funkcja Inicjalizująca bibliotekę

Po nawiązaniu komunikacji klient przy pomocy socketu synchronicznego API wysyła „UserRequest” utworzony na podstawie stringu „AppId” zawierającego informację na temat użytkownika oraz hasła. W odpowiedzi do aplikacji zewnętrznej zostanie wysłany komunikat FIXML „UserResponse”. Informacja zostanie przetworzona na odpowiedni kod „ResultCode”.

```
typedef int ResultCode; // >0 - ok , ResultCode<0 - error
NOL3Client::ResultCode NOL3Client::Initialize
(
    const char* AppId
);
```

5.1.1 Inicjalizacja funkcji zwrotnej statusu aplikacji NOL3

Funkcja zwrotna powinna być ustawiana przed funkcją Initialize. Pozwoli to otrzymać informację o stanie aplikacji NOL3. Nie ustawienie tej funkcji nie spowoduje żadnego błędu. Argumentem funkcji jest zmienna typu int która przyjmuje następujące wartości:

- „1” – aplikacja NOL3 została zamknięta przy działającym interfejsie,
- „2” – aplikacja NOL3 jest offline, status zwracany przy inicjalizacji oraz przy zmianie stanu aplikacji NOL3,
- „3” – aplikacja NOL3 jest online, status zwracany przy inicjalizacji oraz przy zmianie stanu aplikacji NOL3,

„4” – aplikacja NOL3 nie jest uruchomiona. Kod zwracany przy próbie podłączenia,

„5” – aplikacja NOL3 przeszła w tryb bez składania zleceń.

Gdy nie chcemy otrzymywać informacji należy ustawić wskaźnik funkcji zwrotnej na NULL. Wskaźnik do funkcji zwrotnej jest zerowany przy wywołaniu funkcji shutdown().

```
NOLClnet::ResultCode NOLClient::SetCallbackStatus(void(*ptr)(int));
```

5.1.2 Funkcja inicjalizująca wskaźnik do listy papierów

Pamięć dla listy papierów będzie alokowana w funkcji GetTickers. Funkcja InitTickers zwraca wskaźnik, który będzie wskazywał na strukturę zawierającą informację na temat listy papierów. Funkcję InitTickers należy wywołać przed funkcją GetTickers, dla której argumentem jest wskaźnik zwrócony przez InitTickers. Jeśli tego nie uczynimy, funkcja GetTickers nie załokuje pamięci oraz zwróci numer błędu w ResultCode. W przypadku niezainicjalizowanego interfejsu funkcja zwróci NULL.

```
NOL3Client::NOLTickers* NOL3Client::InitListTickers();
```

5.1.3 Funkcja importująca listę papierów

Funkcja służy do pobierania listy papierów wartościowych z serwera dystrybucyjnego poprzez socket synchroniczny. Możliwe jest pobranie jednego papieru, papierów w zależności od kodu rynku lub grupy lub całą listę instrumentów w zależności od ustawień parametrów In_ticker oraz Typ. Umożliwi to bezbłędne wybranie przez użytkowników zewnętrznych aplikacji odpowiednich tickerów. Struktura NolTicker zawiera pełną nazwę, isin papieru, CFI, kod rynku oraz grupę. W strukturze NolTickers znajduje się wskaźnik do tablicy struktur ptrtickerslist oraz rozmiar tej tablicy. W funkcji GetTickers przy pomocy wskaźnika ptrtickers→ptrtickerslist alokowana jest pamięć o rozmiarze ptrtickers→size.

```
typedef struct _NolTicker_
{
    char Isin[13];           // International Securities Identifying Number
    char Name[21];           // Full name of the ticker
    char MarketCode[3];      // Market code
    char CFI[7];             // CFI
    char Group[3];           // ticker's group
} NolTicker;
```

```
typedef struct _NolTickers_
{
    NolTicker* ptrtickerslist; // pointer to list of tickers
    int size;                  // size of list
} NolTickers;
```

```
typedef enum (UndefList=-1,All,Symbol,ISIN,CFI,MarketCode) TypeOfList;

NOL3Client::ResultCode _stdcall GetTickers
{
    NolTickers* ptrtickers,    // pointer to list of tickers
    TypeofList Typ=UndefList,
    NolTicker* in_ticker=NULL
} NolTickers;
```

5.1.4 Zwolnienie struktury danych

Funkcja zwalniana pamięć do przechowywania pełnej listy papierów wartościowych. Dostęp do zaalokowanej pamięci odbywa się przez wskaźnik strtickerslist → ptrtickerslist.

```
NOLClient::ResultCode NOLClient::ReleaseTickers
(
    NolTickers* ptrtickers
);
```

5.1.5 Funkcje modyfikujące filtr papierów w aplikacji zewnętrznej lub NOL3

Aby otrzymywać informacje o notowaniach online powinno się, przed wywołaniem poniższych funkcji, zainicjalizować funkcję zwrrotną. Biblioteka zawiera funkcję dodawania papierów do filtra „AddToFilter” oraz funkcję „RemFromFilter”, którą należy użyć w przypadku kolejnego dodania papierów do filtra, gdy ten nie jest pusty. (funkcja w pierwszej czyści filtr a następnie dodaje papiery). Argumenty „TickersToAdd” oraz „TickersToRem” są stringami mogącym reprezentować numery ISIN (dla Flush równe false), nazwy lub skróty (dla Flush równe true) oddzielone separatorem np. „;”. Funkcja zwrrotna jest wywoływana w przypadku zmian na GPW lub zmian w filtrze. W tej funkcji wykorzystany jest komunikat „MarketDataRequest”. Komunikat będzie przekazywany przez socket synchroniczny. W odpowiedzi zostanie wysłany komunikat „MarketDataSnapshotFullRefresh”, który zostanie przetworzony na odpowiedni kod „ResultCode”. Nie ma możliwości ustawiania ilości otrzymywanych ofert. Maksymalną ilością ofert jest pięć dla użytkowników który mają cały arkusz lub 5 ofert.

```
NOL3Client::ResultCode NOL3Client::AddToFilter
(
    const char* TickersToAdd,bool Flush = false
);
```

```
NOL3Client::ResultCode NOL3Client::RemFromFilter
(
    const char* TickersToRem,bool Flush = false
);
```

5.1.6 Usunięcie papierów z filtra

Funkcja usuwa wszystkie papiery w filtrze znajdującym się w aplikacji NOL3. W funkcji zostanie wykorzystany komunikat „MarketDataRequest”. Komunikat jest przekazywany przez socket synchroniczny. W odpowiedzi zostanie wysłany komunikat „MarketDataSnapshotFullRefresh”, który zostanie przetworzony na „ResultCode”.

```
NOL3Client::ResultCode NOL3Client::ClearFilter();
```

5.1.7 Zapytanie o stan sesji

Funkcja wysyła żądanie otrzymywania/wstrzymania informacji na temat statusu oraz fazy. Funkcja ta powinna być wywołana przed dodaniem papierów do filtra. Informacja o statusie oraz fazie jest przekazywana wraz z image (z nola lub serwera), więc brak wywołania tej funkcji z argumentem true spowoduje brak informacji na ten temat. Komunikat jest przekazywany przez socket synchroniczny. W odpowiedzi zostanie wysłana kod z informacją o poprawności wykonania operacji. Kanąłem asynchronicznym będą napływać informację o fazie sesji oraz statusie.

```
NOL3Client::ResultCode NOL3Client::SetTradingSess(bool set);
```

5.1.8 Inicjalizacja funkcji zwrotnej dotyczącej danych online

Funkcja ustawia metodę do zwrotnego przekazywania danych o notowaniach w czasie rzeczywistym. Funkcja zwrotna wywoływana jest asynchronicznie z poziomu biblioteki. Funkcja odbiera komunikaty FIXML z socketa asynchronicznego: „MarketDataIncrementalRefresh”, zawierające informacje na temat walorów znajdujących się w filtrze. Komunikaty zostaną przetwarzane na strukturę „NoIRecentInfo”. Czas ostatniej transakcji jest zapisywany w postaci stringu.

```
NOL3Client::ResultCode NOL3Client::SetCallback
(
    void (*ptrcallback) (NOL3Client::NolRecentInfo*)
);
```

Zmienna BitMask umieszczona w strukturze „NoRecentInfo” informuje, które zmienne są aktywne:

```
Int BitMask  
    b31. . . b22.....  
Xxxx xx w u t s r p o n m l k j i h g f e d c b a
```

Każda oferta jest zapisywana w oddzielnym obiekcie struktury `NoIBidAskTbl`

```

struct _NolBidAskTbl_
{
    int depth;           // position of offer
    int side;            // side 1- bid, 2-ask

```

```
double price;      // price of offer
int size;          // volume of offer
int amount;        // amount offers
};
```

W strukturze NolRecentInfo zostanie umieszczony wskaźnik do struktury, która zawiera rozmiar oraz tabelę ofert NolBidAskStr.

```

struct _NolBidAskStr_
{
    int offersize;
    NolBidAskTbl* bidask_table;
}NolBidAskStr;

```

```
Int BitMask  
      b31. . . b22.....  
Xxxx xx w u t s r p o n m l k j i h g f e d c b a
```

```

struct _NolRecentInfo_
{
    int BitMask;           // mask of data
    NollTicker ticker;     // ticker
    double ValoLT;         // a(flag in BitMask) - value of last transaction/price
    int VoLT;              // b - volume of last transaction
    char ToLT[9];          // c - time of last transaction for example "2006-11-30 11:12:23"
    double Open;           // d - open price
    double High;           // e - current high price
    double Low;            // f - current low price
    double Close;          // g - close price
    double Bid;            // h - the best bid price
    double Ask;            // i - the best ask price
    int BidSize;           // j - size of the best bid
    int AskSize;           // k - size of the best ask
    int TotalVolume;       // l - total volume
    double TotalValue;     // m - total value
    int OpenInterest;      // n - number of open intrrests
    int Phase;             // o - ticker phase
    char Status[3];        // p - ticker status
    int BidAmount;         // r - amount of bid

```

```

    int AskAmount;           // s - amount of ask
    double OpenValue;        // t - open value
    double CloseValue;       // u - close value
    double ReferPrice;       // w - reference price
    NolBidAskStr offers;     // y - offers
    int Error;               // z - error
} NolRecentInfo;

```

Example:

```

NolRecentInfo new(48,"PLTLKPL00017","TPS.A.",xx,xx,xx,xx,22.5,19.2,xx,xx...);
// xx- doesn't matter , 48 - ...0011 0000- fifth and sixth data are current

```

5.1.9 Inicjalizacja funkcji zwrotnej dotyczącej statusu zleceń

Funkcja ustawia metodę do zwrotnego przekazywania danych o statusie zleceń. Funkcja zwrotna wywoływana jest asynchronicznie z poziomu biblioteki. Funkcja odbiera komunikat FIXML z socketa asynchronicznego „ExecutionReport” i przetwarza je na strukturę „NolOrderReport”. Komunikatem dotyczącym statusu zleceń jest komunikat FIXML „ExecutionReport”.

Komunikat ten dotyczy:

- Potwierdzenie odbioru zlecenia,
- Potwierdzenie zmian w istniejącym zleceniu (anulata/modyfikacje),
- Przekazanie informacji o realizacji (wypełnieniu) zlecenia,
- Przekazanie statusu zlecenia,
- Odrzucenie zlecenia,
- Przekazanie informacji o prowizjach,

Dla negatywnej odpowiedzi na żądanie anulowania lub modyfikacji zlecenia zastosowany jest komunikat FIXML „OrderCancelReject”. System zewnętrzny nie będzie miał możliwości wysłania żądania do serwera o informację o zleceniu/zleceniach.

```

NOL3Client::ResultCode NOL3Client::SetCallbackOrder
(
void (*ptrcallbackorder) ( NOL3Client:NolOrderReport* )
);
typedef struct _NolOrderReport_
{
    int BitMask;           // BitMask w ich data are active
    char ID[];             // bit 0 order ID from library
    char OrdID[];          // bit 1 of BitMask order ID
    char OrdID2[];         // bit 2 secondary order ID
    char StatReqID[];       // bit 3 ID of OrderStatusRequest
    char ExecID[];         // bit 4 transaction ID

```



```

    char ExecTyp;           // bit 5 type of execution
    char Stat;              // bit 6 order status
    int RejRsn;             // bit 7 reject reason
    char Acct[];            // bit 8 account
    NOLTicker ticker;       // bit 9 name, bit 10 isin of ticker structure
    char Side[2];           // bit 11
    int Qty;                // bit 12 Quantity order
    char OrdTyp[2];         // bit 13 order type
    float Px;               // bit 14 price
    float StopPx;           // bit 15 stop price
    char Ccy[];             // bit 16 currency
    char TmInForce[2];      // bit 17 specifies how long the order remains in effect
    char ExpireDt[9];       // bit 18 date of order expiration
    float LastPx;           // bit 19 price of this last fill
    int LastQty;            // bit 20 Quantity bought/sold on this last fill
    int LeavesQty;          // bit 21 Quantity open for further execution
    int CumQty;             // bit 22 Total Quantity
    float TxnTm[18];        // bit 23 time of execution/order creation
    float Comm;             // bit 24 Commission
    float NetMny;           // bit 25 Total amount due as the result of the transaction
    int MinQty;             // bit 26 Minimum quantity of an order to be executed
    int DisplayQty;         // bit 27 the quantity to be displayed
    float TrgrPx;           // bit 28 the price at which the trigger should hit
    char DefPayTp[2];       // bit 29 Deferred Payment Type
    char BizRejRsn;         // bit 30
    char Txt[160];          // bit 31
} NOLOrderReport;

```

5.1.10 Inicjalizacja funkcji zwrotnej dotyczącej wyciągu

Funkcja ustawia metodę do zwrotnego przekazywania danych o wyciągu. Funkcja zwrotna wywoływana jest asynchronicznie z poziomu biblioteki. Funkcja odbiera nowy komunikat z socketa asynchronicznego „Statement”. System zewnętrzny nie będzie miał możliwości wysłania żądania do serwera o wyciąg. Wszystkie żądania będzie obsługiwać w NOL3 na podstawie danych przychodzących z serwera asynchronicznie (na podstawie image przechowywanego w strukturach pamięci NOL3). Wyciąg będzie propagowany przez API wtedy, kiedy przychodzi do modułu transakcyjnego NOL3. Komunikat będzie przetwarzany na strukturę „NOLAggrStatement”.

```

NOL3Client::ResultCode NOL3Client::SetCallbackAccount
(
    void (*ptrcallbackaccount) (NOL3Client:: NOLAggrStatement* )
);

```

```
typedef struct _NolAggrStatement_  
{  
    NolStatement* ptrstate;          // pointer to statement  
    int size;                        // number of accounts  
} NolAggrStatement;  
  
typedef struct _NolStatement_  
{  
    char name[13];                   // account  
    NolFund* ptrfund;               // pointer to table of fundations  
    Int sizefund;                   // size of fundations table  
    NolPos* ptrpos;                 // pointer to table of acctivities  
    Int sizepos;                    // size of acctibities table  
}  
  
typedef struct _NolPos_  
{  
    char isin[13];                   // isin of activity  
    int acc110;                      // amount of acctivity  
    int acc120;                      // amount of acctivity  
}  
  
Typedef struct _NolFund_  
{  
    char name[30];                   // name of foundation  
    char value[30];                 // name of value  
}
```

5.1.11 Inicjalizacja funkcji zwrotnej dotyczącej wizjera

Funkcja ustawia metodę do zwrotnego przekazywania danych dla wizjera. Funkcja zwrotna wywoływana jest asynchronicznie z poziomu biblioteki. Funkcja odbiera komunikaty FIXML „News” z socketa asynchronicznego i przetwarza je na tekst.

```
NOLClient::ResultCode NOL3Client::SetCallbackOutlook  
(  
    void (*ptrcallbackoutlook)( const char*)  
);
```

5.1.12 Inicjalizacja funkcji zwrotnej dotyczącej opóźnień

Funkcja ustawia metodę do zwrotnego przekazywania informacji na temat opóźnienia. Funkcja zwrotna wywoływana jest asynchronicznie z poziomu biblioteki. Funkcja odbiera komunikat FIXML „ApplicationMessageReport” z socketa asynchronicznego. Funkcja zwrotna przyjmuje na wejściu zmienną reprezentującą wartość opóźnienia.

```
NOLClient::ResultCode NOL3Client::SetCallbackDelay
(
    void (*ptrcallbackdelay)(float)
);
```

5.1.13 Funkcja umożliwiająca złożenie, modyfikację, anulację zlecenia oraz otrzymanie informacji o statusie zlecenia

Komunikacja odbywać się będzie przy pomocy połączenia synchronicznego. Do API będzie wysyłany odpowiednio komunikat FIXML „NewOrderSingle”, „OrderCancelReplaceRequest”, „OrderCancelRequest” oraz „OrderStatusRequest” utworzony na podstawie struktury „NolOrderRequest”. W strukturze należy ustawić odpowiednią wartość zmiennej BitMask informując które pola są aktywne. W odpowiedzi do biblioteki zostanie wysłany kod informujący o poprawności operacji oraz jeśli zajdzie taka potrzeba struktura NolOrderReport*. Pozostałe informacje np. o statusie zlecenia będą napływały kanałem asynchronicznym.

```
typedef enum (Undef = -1, NewOrder, ModOrder, DelOrder, StatOrder) OrderType;

NOL3Client::ResultCode NOL3Client::APIOrderRequest
(
    NOL3Client::NolOrderRequest* order, NolOrderReport* , OrderType Typ
);
```

```
int BitMask
b31...b19.....b0
xxxxxx u t s r p o n m l k j I h g f e d c b a

typedef struct _NolOrderRequest_
{
    Int BitMask;
    char OrigID[10];    //a    order ID from library
    char OrdID[10];     //b    secondary order ID from library
    char OrdID2[10];    //c    order ID
    char Acct[17];      //d    account
    int MinQty;         //e    minimum quantity of an order to be executed
    int DisplayQty;     //f    the quantity to be displayed
    NolTicker ticker;   //g,h   ticker structure, g-isin, h-name/alias
    char Side;          //i    buy/sale
    int Qty;            //j    quantity ordered
    char OrdTyp[2];     //k    order type
    float Px;           //l    price
    float StopPx;       //m    stop price
    char Ccy[6];        //n    currency
    char TmInForce[2];  //o    specifies how long the order remains in effect
```

```
char ExpireDt[9]; //p    date of order expiration
float TrgrPx;      //r    the price at which the trigger should hit
char DefPayTyp[2]; //s    Deferred Payment Type
char SessionDt[9]; //t    Session Date
} NolOrderRequest;
```

5.1.14 Odczyt opisu kodu błędu

Funkcja zwraca opis tekstowy do kodu, który podany jest jako argument. Deskryptor kodów zostanie dostarczony wraz z implementacją biblioteki.

```
const char* NOL3Client::GetResultCodeDesc
(
    NOL3Client::ResultCode code          /* code returned by function */
);
```

5.1.15 Odczyt wersji biblioteki

Funkcja zwracająca numer wersji biblioteki.

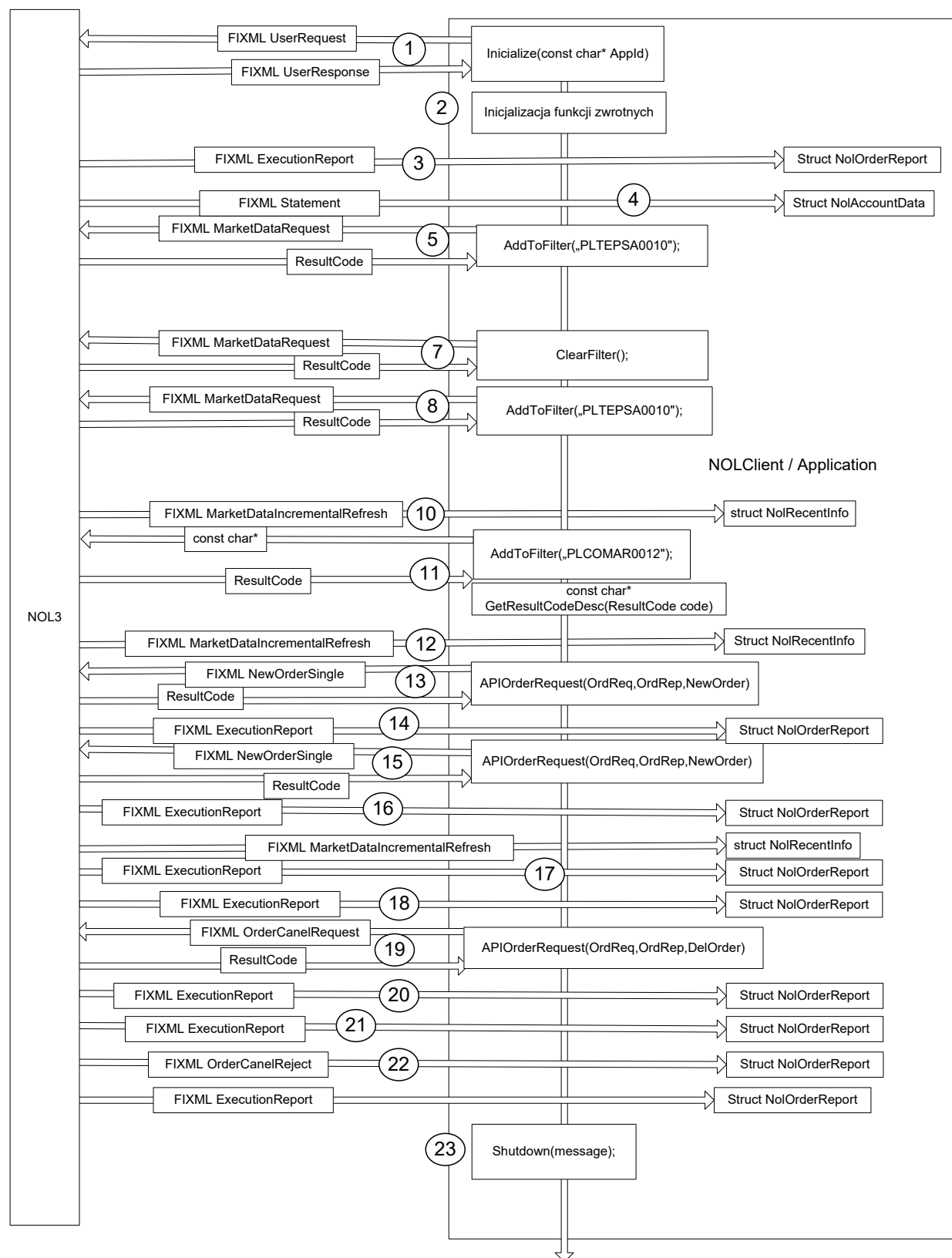
```
const char* NOL3Client::GetVersion(void);
```

5.1.16 Zakończenie pracy z biblioteką

Funkcja zamykająca komunikację z aplikacją NOL3, zwalnia zaalokowane zasoby.

```
NOL3Client::ResultCode NOL3Client::Shutdown(void);
```

5.2 Przykład działania biblioteki



- 1) Inicjalizacja biblioteki, logowanie,
- 2) Ustawienie funkcji zwrotnych do otrzymywania danych online, wyciągu, statusu zleceń, informacje dla wyciągu, wizjera oraz sesji,
- 3) Komunikat na temat bieżących zleceń,
- 4) Komunikat dla wyciągu,
- 5) Dodanie papieru do filtra,
- 6) Usunięcie wszystkich papierów z filtra,
- 7) Dodanie papieru do filtra,
- 8) Komunikat o papierze znajdujących się w filtrze,
- 9) Dodanie papieru do filtra z automatycznym zatwierdzeniem filtra, sprawdzenie kodu „NolRecentInfo”,
- 10) Wywołanie funkcji zwrotnej spowodowane zmianą w filtrze, komunikat zawierający informację o zmianach,
- 11) Złożenie zlecenia, komunikat „NewOrderSingle”,
- 12) Otrzymanie komunikatu z DMBOŚ o przyjęciu zlecenia, komunikat „ExecutionReport”,
- 13) Złożenie zlecenia, komunikat „NewOrderSingle”,
- 14) Otrzymanie komunikatu z DMBOŚ o przyjęciu zlecenia, komunikat „ExecutionReport”,
- 15) Otrzymanie komunikatu z DMBOŚ o przyjęciu zlecenia na GPW, komunikat „ExecutionReport”,
- 16) Otrzymanie komunikatu z DMBOŚ o złożeniu zlecenia na GPW, komunikat „ExecutionReport”,
- 17) Anulowanie zlecenia, komunikat „OrderCancelRequest”,
- 18) Otrzymanie komunikatu z DMBOŚ o przyjęciu anulaty zlecenia, komunikat „ExecutionReport”,
- 19) Otrzymanie komunikatu z GPW o odrzuceniu anulaty, komunikat „OrderCancelReject”,
- 20) Otrzymanie komunikatu z GPW o dokonaniu transakcji, komunikat „ExecutionReport”,
- 21) Zamknięcie aplikacji, zwolnienie zaalokowanych zasobów.

5.3 Deskryptor błędów – biblioteka

5.3.1 INITIALIZE

- 5: "inicializing lib is finished(five offers package)";
- 1: "lib is initalized";
- 0: "inicializing lib is finished";
- 2: "Library winsock2 not found";
- 3: "Unsupported winsock2 library";
- 4: "Sync Socket not connected";
- 7: "Async Socket not connected";
- 13: "Init Message not initialized";
- 14: "Sending Message is empty";
- 15: "Error sending message - server socket is closed";

- 16: "Error Sending message - server socket is closed";
- 17: "Root xml error";
- 19: "Attribute error";
- 20: "Length error in recieved message";
- 21: "Error recieved message";
- 57: "unknown user";
- 58: "unknown password";
- 79: "unknown tag in message";
- 80: "wrong id in UserReq";
- 68: "BizMsgRej BizRejRsn: 0";
- 69: "BizMsgRej BizRejRsn: 1";
- 70: "BizMsgRej BizRejRsn: 2";
- 71: "BizMsgRej BizRejRsn: 3";
- 72: "BizMsgRej BizRejRsn: 4";
- 74: "BizMsgRej BizRejRsn: 5";
- 75: "BizMsgRej BizRejRsn: 6";
- 76: "BizMsgRej BizRejRsn: 7";
- 81: "BizMsgRej BizRejRsn: 18";
- 82: "BizMsgRej unknown BizRejRsn";
- 83: "Unknown status";
- 84: "Unknown ReqRejResn";
- 86: "Could not open registry for sync";
- 87: "Registry for sync isnt set";
- 88: "Registry for sync are null";
- 90: "Could not open registry for async";
- 91: "Registry for async isnt set";
- 92: "Registry for async is null";

5.3.2 ADDTOFILTER

- 6: "add to filter";
- 4: "Sync Socket not connected";
- 14: "Sending Message is empty";
- 15: "Error sending message - server socket is closed";
- 16: "Error Sending message - server socket is closed";
- 17: "Root xml error";
- 19: "Attribute error";
- 20: "Length error in recieved message";
- 21: "Error recieved message";
- 22: "Lib is not inicialized";
- 33: "wrong parameter maxNumInstrInAPITab";
- 34: "error with attribute action or ticker";

-56: "No securities to add/remove";
-59: "wrong id in MktDataReq";
-60: "unsupported instrument";
-61: "unknown MarketDepth";
-62: "unknown action(SubReqTyp)";
-68: "BizMsgRej BizRejRsn: 0";
-69: "BizMsgRej BizRejRsn: 1";
-70: "BizMsgRej BizRejRsn: 2";
-71: "BizMsgRej BizRejRsn: 3";
-72: "BizMsgRej BizRejRsn: 4";
-74: "BizMsgRej BizRejRsn: 5";
-75: "BizMsgRej BizRejRsn: 6";
-76: "BizMsgRej BizRejRsn: 7";
-81: "BizMsgRej BizRejRsn: 18";
-82: "BizMsgRej unknown BizRejRsn";
-83: "Unknown status";
-84: "Unknown ReqRejResn";

5.3.3 REMFROMFILTER

7: "remove from filter";
-4: "Sync Socket not connected";
-14: "Sending Message is empty";
-15: "Error sending message - server socket is closed";
-16: "Error Sending message - server socket is closed";
-17: "Root xml error";
-19: "Attribute error";
-20: "Length error in recieved message";
-21: "Error recieved message";
-22: "Lib is not inicialized";
-33: "wrong parameter maxNumInstrInAPITab";
-34: "error with attribute action or ticker";
-56: "No securities to add/remove";
-59: "wrong id in MktDataReq";
-60: "unsupported instrument";
-62: "unknown action(SubReqTyp)";
-61: "unknown MarketDepth";
-68: "BizMsgRej BizRejRsn: 0";
-69: "BizMsgRej BizRejRsn: 1";
-70: "BizMsgRej BizRejRsn: 2";
-71: "BizMsgRej BizRejRsn: 3";
-72: "BizMsgRej BizRejRsn: 4";

-74: "BizMsgRej BizRejRsn: 5";
-75: "BizMsgRej BizRejRsn: 6";
-76: "BizMsgRej BizRejRsn: 7";
-81: "BizMsgRej BizRejRsn: 18";
-82: "BizMsgRej unknown BizRejRsn";
-83: "Unknown status";
-84: "Unknown ReqRejResn";

5.3.4 CLEARFILTER

8: "clear filter";
-4: "Sync Socket not connected";
-14: "Sending Message is empty";
-15: "Error sending message - server socket is closed";
-16: "Error Sending message - server socket is closed";
-17: "Root xml error";
-19: "Attribute error";
-20: "Length error in recieved message";
-21: "Error recieved message";
-22: "Lib is not inicialized";
-59: "wrong id in MktDataReq";
-60: "unsupported instrument";
-61: "unknown MarketDepth";
-62: "unknown action(SubReqTyp)";
-68: "BizMsgRej BizRejRsn: 0";
-69: "BizMsgRej BizRejRsn: 1";
-70: "BizMsgRej BizRejRsn: 2";
-71: "BizMsgRej BizRejRsn: 3";
-72: "BizMsgRej BizRejRsn: 4";
-74: "BizMsgRej BizRejRsn: 5";
-75: "BizMsgRej BizRejRsn: 6";
-76: "BizMsgRej BizRejRsn: 7";
-81: "BizMsgRej BizRejRsn: 18";
-82: "BizMsgRej unknown BizRejRsn";
-83: "Unknown status";
-84: "Unknown ReqRejResn";

5.3.5 SHUTDOWN

17: "shutdown without authorization";
10: "shutdown";
-14: "Sending Message is empty";
-15: "Error sending message - server socket is closed";

- 16: "Error Sending message - server socket is closed";
- 17: "Root xml error";
- 19: "Attribute error";
- 20: "Length error in recieved message";
- 21: "Error recieved message";
- 22: "Lib is not inicialized";
- 68: "BizMsgRej BizRejRsn: 0";
- 69: "BizMsgRej BizRejRsn: 1";
- 70: "BizMsgRej BizRejRsn: 2";
- 71: "BizMsgRej BizRejRsn: 3";
- 72: "BizMsgRej BizRejRsn: 4";
- 74: "BizMsgRej BizRejRsn: 5";
- 75: "BizMsgRej BizRejRsn: 6";
- 76: "BizMsgRej BizRejRsn: 7";
- 81: "BizMsgRej BizRejRsn: 18";
- 82: "BizMsgRej unknown BizRejRsn";
- 83: "Unknown status";
- 84: "Unknown ReqRejResn";

5.3.6 SETCALLBACK, SETCALLBACKACCOUNT, SETCALLBACKORDER, SETCALLBACKOUTLOOK, SETCALLBACKDELAY, SETCALLBACKSTATUS

- 13: "Initializing callback function";
- 15: "callback is already initialized";
- 16: "callback function is deactivated";

5.3.7 APIORDERREQUEST

- 18: "order is submitted";
- 4: "Sync Socket not connected";
- 14: "Sending Message is empty";
- 15: "Error sending message - server socket is closed";
- 16: "Error Sending message - server socket is closed";
- 17: "Root xml error";
- 19: "Attribute error";
- 20: "Length error in recieved message";
- 21: "Error recieved message";
- 22: "Lib is not inicialized";
- 55: "Only one function may be called";
- 63: "Unallocated structure - order";
- 64: "Unallocated structure - report";
- 65: "unknown action - order";
- 66: "Error in BitMask - order";

- 67: "Error in order - info in Txt";
- 68: "BizMsgRej BizRejRsn: 0";
- 69: "BizMsgRej BizRejRsn: 1";
- 70: "BizMsgRej BizRejRsn: 2";
- 71: "BizMsgRej BizRejRsn: 3";
- 72: "BizMsgRej BizRejRsn: 4";
- 74: "BizMsgRej BizRejRsn: 5";
- 75: "BizMsgRej BizRejRsn: 6";
- 76: "BizMsgRej BizRejRsn: 7";
- 81: "BizMsgRej BizRejRsn: 18";
- 82: "BizMsgRej unknown BizRejRsn";
- 83: "Unknown status";
- 84: "Unknown ReqRejResn";

5.3.8 Błędy zwracane w NolRecentInfo

- 28: "wsa_wait_failed";
- 29: "socket error";
- 30: "error event read";

5.3.9 SetTradingSess

- 19: "SetTradingSess function is executed";
- 19: "Attribute error";
- 22: "Lib is not inicialized";
- 55: "Only one function may be called";
- 68: "BizMsgRej BizRejRsn: 0";
- 69: "BizMsgRej BizRejRsn: 1";
- 70: "BizMsgRej BizRejRsn: 2";
- 71: "BizMsgRej BizRejRsn: 3";
- 72: "BizMsgRej BizRejRsn: 4";
- 74: "BizMsgRej BizRejRsn: 5";
- 75: "BizMsgRej BizRejRsn: 6";
- 76: "BizMsgRej BizRejRsn: 7";
- 81: "BizMsgRej BizRejRsn: 18";
- 82: "BizMsgRej unknown BizRejRsn";
- 83: "Unknown status";
- 85: "TradSesStatReq error";

5.3.10 ReleaseTickersList

- 12: "no memory to release"
- 3: "memory for tickers list is released"
- 22: "Lib is not inicialized";

5.3.11 GetTickers

- 4: "Sync Socket not connected";
- 14: "Sending Message is empty";
- 15: "Error sending message - server socket is closed";
- 16: "Error Sending message - server socket is closed";
- 17: "Root xml error";
- 19: "Attribute error";
- 20: "Length error in recieved message";
- 21: "Error recieved message";
- 22: "Lib is not inicialized";
- 23: "Tickers pointer not inicialized";
- 24: "Memory for message is not initialized";
- 55: "Only one function may be called";
- 93: "There is no instrument";
- 94: "Set Symbol of instrument";
- 95: "Set isin of instrument";
- 96: "Set code market";
- 97: "Set CFI";
- 98: "No instrument found";
- 99: "Instrument data unavalible";
- 100: "Requested instrument not supported";
- 101: "Unknown error in RequestList";