

# **Eksploracja technologii JXTA**

**Wojciech Jaworski**

**Bartosz Miłosierny**

<u>Wstęp – idee.....</u>	<u>3</u>
<u>Czym jest JXTA?.....</u>	<u>3</u>
<u>Architektura.....</u>	<u>5</u>
<u>Komponenty JXTA.....</u>	<u>6</u>
<u>Kluczowe aspekty architektury.....</u>	<u>6</u>
<u>Terminologia i podstawowe koncepcje.....</u>	<u>6</u>
<u>Węzły.....</u>	<u>6</u>
<u>Grupy węzłów.....</u>	<u>7</u>
<u>Usługi.....</u>	<u>7</u>
<u>Moduły.....</u>	<u>7</u>
<u>Potoki.....</u>	<u>7</u>
<u>Zaawansowane kanały komunikacji.....</u>	<u>8</u>
<u>Komunikaty.....</u>	<u>8</u>
<u>Anonse.....</u>	<u>8</u>
<u>Bezpieczeństwo.....</u>	<u>9</u>
<u>Architektura sieci.....</u>	<u>9</u>
<u>Protokoły.....</u>	<u>11</u>
<u>Przykłady.....</u>	<u>12</u>
<u>Przykład 1 – wykrywanie węzłów sieci.....</u>	<u>12</u>
<u>Przykład 2 – tworzenie grupy węzłów.....</u>	<u>13</u>
<u>Przykład 3 – usługa potoku.....</u>	<u>13</u>

## **Wstęp – idee**

JXTA™ jest zbiorem otwartych protokołów, które pozwalają dowolnym urządzeniom podłączonym do sieci (od telefonów komórkowych i PDA przez komputery biurkowe aż do superserwerów) występować w charakterze węzła sieci peer-to-peer – to znaczy komunikować się i współpracować z innymi urządzeniami na zasadzie „równy z równym”.

Węzły sieci korzystające z technologii JXTA tworzą wirtualną sieć, w której każdy z węzłów może dokonywać interakcji z innymi węzłami lub zasobami bezpośrednio, nawet w sytuacji, w której niektóre węzły znajdują się za zaporami lub NAT.

Cele technologii JXTA to przede wszystkim:

- Interoperacyjność – współpraca w różnych także heterogenicznych systemach peer-to-peer i społecznościach sieciowych
- Niezależność od platformy – zróżnicowane języki programowania, systemy operacyjne i sieci
- Wszechobecność – możliwość stosowania na dowolnym urządzeniu

JXTA daje użytkownikom i programistom wiele możliwości, z których najważniejsze to:

- Znajdowanie węzłów i zasobów w sieci bez względu na zapory i NAT
- Dzielenie się plikami z kimkolwiek w sieci
- Tworzenie własnych grup węzłów nawet, jeśli fizycznie znajdują się w różnych sieciach
- Zapewnienie bezpiecznej komunikacji między węzłami w sieciach publicznych

Projekt JXTA™ zaczynał jako projekt badawczy zapoczątkowany w Sun Microsystems pod kierownictwem Billa Joy'a i Mike'a Clary'ego i miał na celu zaadresowanie sieci peer-to-peer.

## **Czym jest JXTA?**

JXTA jest zatem otwartą platformą służącą do obliczeń w sieciach zaprojektowaną do pracy na zasadach peer-to-peer (P2P). Jej celem jest rozwinięcie podstawowych modułów i usług umożliwiających tworzenie aplikacji dla grup P2P. Termin „JXTA” (czytamy: „dzaksta”) jest skrótem od angielskiego słowa „juxtapose” oznaczającego „zestawiać obok siebie”. JXTA

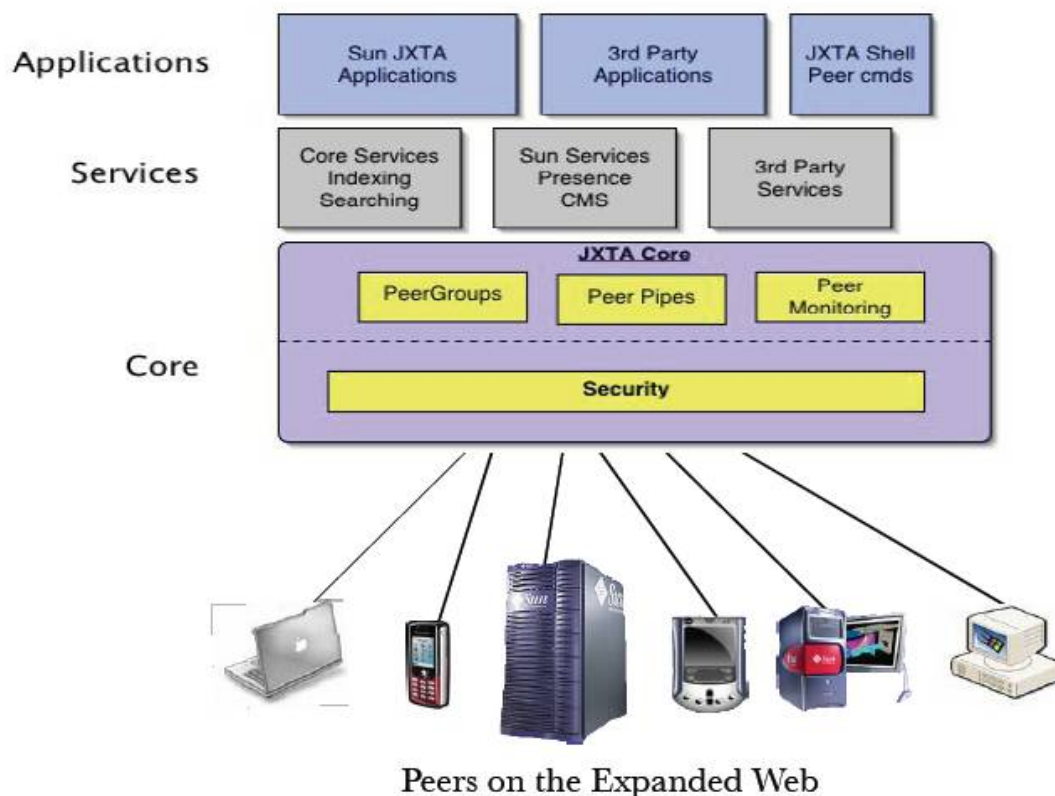
dostarcza wspólnego zbioru otwartych protokołów i open-source'owej implementacji umożliwiającej rozwijanie aplikacji P2P.

Protokoły JXTA standaryzują sposób, w jaki węzły sieci P2P:

- Odkrywają siebie nawzajem
- Samoorganizują się w grupy
- Rozgłaszają i odkrywają usługi sieciowe
- Komunikują się ze sobą
- Monitorują nawzajem

Protokoły JXTA zostały zaprojektowane niezależnie od języków programowania i niezależnie od protokołów transportowych. Protokoły JXTA mogą być z powodzeniem zaimplementowane w takich językach jak: Java, C/C++, Perl i innych, ponad protokołami transportowymi takimi jak: TCP/IP, HTTP, Bluetooth, HomePNA i innymi.

## Architektura



### 1. *Warstwa Platformy (JXTA core)*

Warstwa platformy czyli rdzeń JXTA zawiera minimalny i niezbędny zbiór funkcji, które są wspólne dla modelu P2P. Zawiera mechanizmy kluczowe dla P2P – pozwalające na odkrywanie węzłów i grup, zapewniające transport (także poprzez zapory), umożliwiające tworzenie węzłów i grup, a także funkcje realizujące bezpieczeństwo transmisji.

### 2. *Warstwa Usług*

Warstwa usług zawiera usługi sieciowe, które nie są całkowicie niezbędne do funkcjonowania sieci P2P, ale są często spotykana bądź pożądane w środowisku P2P. Przykłady takich usług to: wyszukiwanie i indeksowanie, katalogowanie, systemy składu, współdzielenie plików, rozproszone systemy plików, agregacja i najmowanie zasobów, tłumaczenie protokołów, uwierzytelnianie i usługi PKI (Public Key Infrastructure)

### 3. *Warstwa Aplikacji*

Warstwa aplikacji zawiera implementacje zintegrowanych aplikacji takich jak komunikatory P2P, współdzielenie zasobów i dokumentów, systemy poczty elektronicznej P2P i tym podobne

## **Komponenty JXTA**

Sieć JXTA składa się z szeregu połączonych węzłów (ang. *peers*). Węzły mają zdolność do samoorganizowania się w grupy, które dostarczają podobnych usług. Węzły JXTA rozgłaszają swoje usługi w dokumentach XML zwanych „anonsami” (ang. *advertisements*). Anonse pozwalają innym węzłom w sieci zdobyć wiedzę niezbędną do połączenia się z usługą i skorzystania z niej.

Węzły JXTA używają potoków (ang. *pipes*) do przesyłania komunikatów do siebie nawzajem. Potoki to asynchroniczny i niekierowany mechanizm transferu używany do komunikacji między serwisami. Komunikaty to proste dokumenty XML, których „koperta” zawierają między innymi trasę, streszczenie komunikatu i dane niezbędne do autoryzacji. Potoki są związane z tak zwanymi punktami terminalnymi (ang. *endpoints*), którymi mogą być: port TCP i odpowiedni adres IP.

## **Kluczowe aspekty architektury**

Trzy kluczowe cechy architektury JXTA odróżniają ją od innych modeli rozproszonych:

- Użycie XML-owych dokumentów do anonsowania i opisywania usług sieciowych
- Abstrakcja połączeń między węzłami (zarówno w kontekście potoków, jak i punktów terminalnych) niezależna od centralnego systemu adresowania takiego jak DNS
- Spójna koncepcja adresowania węzłów (poprzez *peer ID*)

## **Terminologia i podstawowe koncepcje**

### **Węzły**

Węzeł to dowolne urządzenie w sieci implementujące jeden lub więcej protokołów JXTA. Może nim być zarówno superkomputer, terminal jak i urządzenia przenośne typu PDA, telefony komórkowe. Każdy interfejs sieciowy takiego urządzenia, rozgłoszony i znany innym urządzeniom staje się punktem terminalnym (*endpoint*) i może zostać użyty do nawiązania połączenia typu *point-to-point* z innym interfejsem.

### **Grupy węzłów**

Grupą nazywamy każdy zbiór węzłów świadczący określony zbiór usług i posiadający niepowtarzalny identyfikator *group ID*. Węzeł może należeć do dowolnej liczby grup. Domyślną grupą, do której zostaje przyłączony każdy węzeł w czasie startu jest *Net Peer Group*. Grupowanie węzłów pozwala na osiągnięcie wyższego poziomu bezpieczeństwa komunikacji i lepszej specjalizacji.

Każda grupa węzłów świadczy pewien zestaw usług (*group services*). Istnieje podstawowy zestaw usług oferowanych przez większość grup, m.in. usługa odnajdywania węzłów (*discovery service*), usługa pozwalająca na przyłączanie nowych węzłów do grupy (*membership service*), usługa potoków (*pipe service*), umożliwiająca przesyłanie komunikatów.

### **Usługi**

Usługi są podstawowym mechanizmem JXTA umożliwiającym pracę węzłów i komunikację między nimi, są implementacją protokołów JXTA. Usługi dzielimy na te świadczone przez pojedynczy węzeł (*peer services*) i usługi grupowe (*group services*). Usługa pojedynczego

węzła staje się niedostępna w przypadku jego awarii, natomiast pracy usługi grupowej pojedyncze awarie nie zakłóca. Usługi mogą być zainstalowane na danym węźle i startować razem z nim lub być dynamicznie ładowane z sieci.

### **Moduły**

Moduł jest pojęciem reprezentującym dowolny wykonywalny kod, np. klasa Javy, biblioteka DLL, odpowiadający za implementację pewnego zachowania systemu. Moduły mogą np. reprezentować implementację pewnej usługi dla różnych systemów operacyjnych. Moduły są rozsyłane i dynamicznie ładowane podczas pracy węzłów, co umożliwia np. automatyczną konfigurację i zlecenie zadań węzłowi po przyłączeniu się do grupy.

### **Potoki**

Potoki są asynchronicznym i nieukierunkowanym kanałem przesyłania komunikatów i danych (binarnych i tekstowych) pomiędzy węzłami. Interfejs wysyłający dane określamy jako *output pipe*, odbierający to *input pipe*. Potoki są wirtualnym kanałem i w rzeczywistości połączenie może być realizowane przy pomocy kilku pośredniczących punktów terminalnych. Istnieje możliwość wysyłania i odbierania danych przez kilka węzłów za pomocą jednego potoku (*unicast pipes* – jeden odbiorca, *propagate pipes* – wielu odbiorców). Aby mieć pewność, że wysłane dane zostaną dostarczone nadawcy należy użyć potoków bezpiecznych (*secure pipes*).

### **Zaawansowane kanały komunikacji**

- *JxtaSocket* i *JxtaServerSocket* – zapewniają komunikację dwukierunkową i potwierdzanie odbioru, oparte na strumieniach Javy, dziedziczą z klas `java.net.Socket` i `Java.net.ServerSocket`
- *JxtaBiDiPipe* i *JxtaServerPipe* - zapewniają komunikację dwukierunkową i potwierdzanie odbioru, ale oparte o mechanizm potoków a nie strumieni

### **Komunikaty**

Komunikat są obiektami wysyłanymi pomiędzy węzłami, np. przy użyciu potoków. Mogą mieć reprezentację XML-ową lub binarną. Każdy komunikat jest ciągiem elementów określonego typu (np. tekst, klasa Javy), identyfikowanych nazwami. Mechanizm *platform binding* odpowiada za skonwertowanie otrzymanego obiektu do formatu odpowiedniego dla danej platformy.

## **Anonse**

Wszystkie zasoby JXTA (węzły, grupy, potoki i usługi) są reprezentowane przez XML-owe dokumenty zwane anonsami (*advertisements*). Są one używane do dokładnego opisu zasobu i publikowane w sieci, dzięki czemu inne węzły mogą uzyskać informacje o odległych zasobach. Węzeł po odczytaniu anonsu może go scachować na lokalnej maszynie.

Przykładowy anons:

```
<?xml version="1.0"?>
<!DOCTYPE jxta:PipeAdvertisement>
<jxta:PipeAdvertisement xmlns:jxta="http://jxta.org">
  <Id>
    urn:jxta:uuid-
    59616261646162614E504720503250338E3E786229EA460DADC1A176B69B731504
  </Id>
  <Type> JxtaUnicast </Type>
  <Name> TestPipe.end1 </Name>
</jxta:PipeAdvertisement>
```

Anons zawiera m.in. identyfikator zasobu. Standardowo jest on generowany w sposób losowy przez platformę.

## **Bezpieczeństwo**

JXTA oferuje liczne mechanizmy bezpieczeństwa. Każdy węzeł podlega autentykacji i autoryzacji w trakcie logowania się do sieci. Komunikaty wysyłane między węzłami mogą być szyfrowane i podpisane cyfrowo. JXTA będzie też wspierać popularne protokoły IPsec i SSL.

## **Architektura sieci**

Sieć JXTA jest w budowana ad hoc z połączonych węzłów. Połączenia w takiej sieci mogą mieć chwilowy, ulotny charakter, a trasa komunikatu nie jest z góry ustalona. Węzły w każdej chwili mogą dołączyć do sieci lub odłączyć się od niej, zatem trasy mogą ulegać częstym zmianom. Węzły mogą przybierać dowolną formę, jeśli tylko potrafią komunikować się przy użyciu protokołów JXTA. Struktura sieci nie jest narzucana odgórnie przez JXTA Framework, jednak zwykle w praktyce używa się czterech rodzajów węzłów:

- *Podstawowy węzeł krańcowy*

Ten typ węzła potrafi wysyłać i odbierać komunikaty, ale nie dokonuje cache'owania anonsów ani nie przekazuje dalej (nie trasuje) komunikatów pochodzących od innych węzłów. Podstawowych węzłów krańcowych używa się zwykle na urządzeniach o ograniczonych zasobach (np. PDA albo telefony komórkowe).

- *Pełny węzeł krańcowy*

Pełen węzeł krańcowy wysyła i odbiera komunikaty i domyślnie dokonuje cache'owania anonsów. Ten typ węzła odpowiada na żądania wykrywania używając informacji pochodzących z zachowanych w cache'u anonsów, ale nie przekazuje dalej żadnych żądań wykrywania. W typowych sytuacjach większość węzłów to właśnie węzły krańcowe.

- *Węzeł rendezvous*

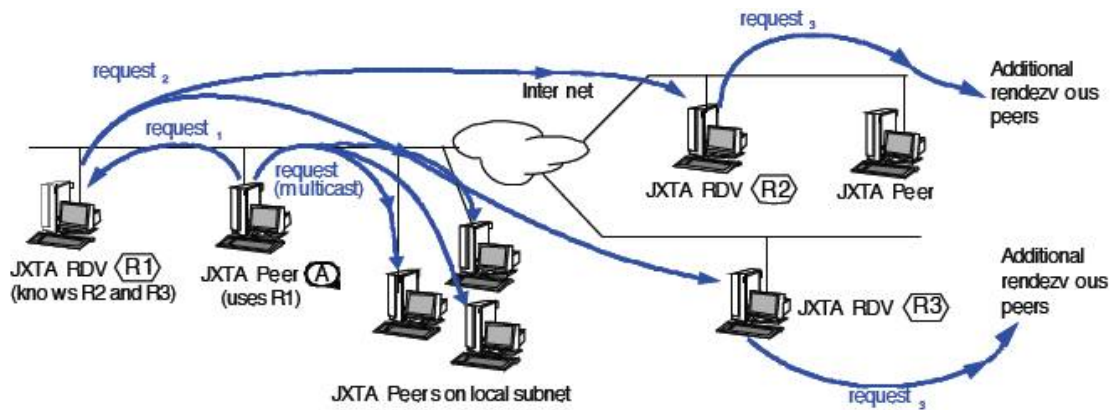
Węzeł rendezvous (fr. *rendezvous* – spotkanie) od pełnego węzła krańcowego różni tym, że przekazuje do innych węzłów żądania wykrywania, umożliwiając innym węzłom zorientowanie się, jakie usługi są udostępniane. Kiedy węzeł A dołącza do grupy automatycznie poszukuje węzła rendezvous. Jeśli żaden taki węzeł nie zostanie znaleziony, węzeł A automatycznie staje się węzłem rendezvous dla tej grupy. Każdy węzeł rendezvous utrzymuje listę innych znanych węzłów rendezvous oraz listę węzłów, które używają tego węzła jako swojego węzła rendezvous.

Każda grupa węzłów utrzymuje własną listę węzłów rendezvous i może posiadać tyle węzłów RDV, ile potrzebuje. Żądania wyszukiwania w danej grupie są widoczne tylko przez te węzły RDV, które są członkami tej grupy.

Węzły krańcowe wysyłają żądania do węzłów RDV, które z kolei przekazują do innych węzłów RDV te żądania, na które nie są w stanie same odpowiedzieć. Proces ten jest kontynuowany dopóki odpowiedź nie zostanie znaleziona albo żądanie umrze. Komunikaty mają domyślnie ustawiony czas życia (TTL) na siedem przeskoków. Przeskoki zwrotne są eliminowane dzięki utrzymywaniu listy krawędzi znajdujących się na trasie komunikatu.

- *Węzły przekaźnikowe*

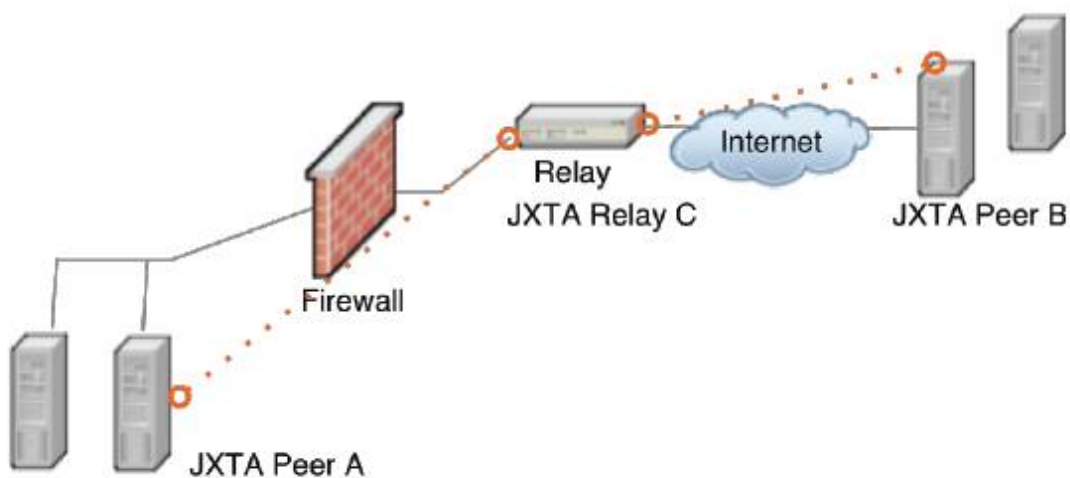
Węzły tego typu posiadają informacje dotyczące tras do innych węzłów i przekazują komunikaty. Węzeł najpierw zagląda do swojego lokalnego cache'u w poszukiwaniu informacji o trasie. Gdy takiej informacji nie znajdzie, wysyła zapytanie do węzła przekaźnikowego o trasę. Węzły przekaźnikowe służą także przekazywaniu komunikatów między węzłami, które nie mogą być bezpośrednio zaadresowane (np. w środowiskach typu NAT), spinając różne fizyczne i/lub logiczne sieci. Usługi charakterystyczne dla węzłów przekaźnikowych czy RDV mogą zostać zaimplementowane na dowolnym węźle. Węzeł może jednocześnie pełnić obie role.



Przykład rozpowszechniania komunikatów w sieci przez węzły RDV.

Komunikacja może też odbywać się pomimo istnienia przeszkód takich jak sieci NAT czy zapory. Aby było to możliwe muszą być spełnione następujące warunki:

- Przynajmniej jeden węzeł w grupie po wewnętrznej stronie zapory musi być znać przynajmniej jeden węzeł poza zapora
- Węzły wewnątrz i na zewnątrz sieci chronionej przez zapora muszą znać siebie nawzajem i muszą obsługiwać protokół HTTP
- Zapora musi pozwalać na transfer przez protokół http



## **Protokoły**

Istnieje 6 protokołów JXTA:

- *Peer Discovery Protocol (PDP)* – używany przez węzły do anonsowania ich własnych zasobów i wykrywania zasobów innych węzłów. Każdy zasób jest opisywany i publikowany za pomocą anonsu.
- *Peer Information Protocol (PIP)* – używany do uzyskiwania informacji kontrolnych (np. czas działania, stan) od węzłów
- *Peer Resolver Protocol (PRP)* – umożliwia węzłom wysyłanie generycznych zapytań to jednego lubi więcej węzłów i odbieranie odpowiedzi (jednej lub wielu). Zapytania mogą zostać przekierowane do wszystkich węzłów w grupie bądź do określonego zbioru węzłów grupy. W przeciwieństwie do PDP i PIP, które umożliwiają wymianę tylko z góry zdefiniowanych komunikatów, ten protokół pozwala na zdefiniowanie i wymianę dowolnej informacji
- *Pipe Binding Protocol (PBP)* – używany przez węzły do ustanowienia wirtualnego kanału komunikacji (czyli potoku). PBP jest używany przez węzeł do zbindowania dwóch lub więcej punktów terminalnych połączenia
- *Endpoint Routing Protocol (ERP)* – służy do znajdowania tras prowadzących do docelowych portów na innych węzłach. Informacja o trasie zawiera sekwencję identyfikatorów węzłów przekaźnikowych, które mogą zostać użyte, aby przesłać komunikat. (Na przykład, komunikat może być wysłany do węzła A, następnie przesłany do węzła B, który ostatecznie skieruje go do docelowego węzła.)
- *Rendezvous Protocol (RVP)* – to mechanizm dzięki, któremu węzły mogą być dostawcami lub odbiorcami usługi rozpowszechniania. Węzły zorganizowane w grupę są albo węzłami RDV albo nasłuchują tych węzłów. Protokół RVP pozwala wysyłać komunikaty do wszystkich nasłuchujących instancji serwisu. RVP jest używany przez Peer Resolver Protocol oraz przez Pipe Binding Protocol do propagowania komunikatów w grupie.

Wszystkie protokoły JXTA są asynchroniczne i oparte na modelu zapytanie/odpowiedź. Węzeł używa ich do wysłania zapytania do pewnej liczby węzłów w grupie. W rezultacie może uzyskać zero, jedną lub więcej odpowiedzi. Na przykład, jakiś węzeł używając protokołu PDP zapytuje o wszystkie znane węzły w supergrupie Neet Peer Group. Typowa

sytuacja w tym przypadku to uzyskanie wielu odpowiedzi od różnych węzłów. Inny przykład: węzeł w grupie poszukuje konkretnego potoku nazwanego „GorskiPotok”. Jeśli potok o tej nazwie nie zostanie znaleziony, uzyskamy dokładnie zero odpowiedzi na to zapytanie.

Węzły JXTA nie są zobowiązane do implementowania wszystkich 6 protokołów – mogą implementować jedynie te, których będą używać. Konkretna implementacja technologii JXTA – platforma JXTAJ2SE wspiera wszystkie sześć. W tej platformie dostępne jest API napisane w języku Java, dzięki któremu można skorzystać w pełni z funkcjonalności protokołów (np. można wykrywać węzły sieci albo dołączyć do grupy).

## **Przykłady**

### **Przykład 1 – wykrywanie węzłów sieci**

Najważniejsze cechy programu:

- Działanie w domyślnej grupie *NetPeerGroup*
- Implementacja interfejsu *DiscoveryListener* (model event-driven)
- Pobieranie anonsów od znalezionych węzłów

### **Przykład 2 – tworzenie grupy węzłów**

Najważniejsze cechy programu:

- Tworzenie nowego anonsu dla nowo tworzonej grupy (w sposób programowalny, nie deklaratywny) z wykorzystaniem jednej z gotowych implementacji
- Publikowanie swojego anonsu
- Dołączanie do nowopowstałej grupy

### **Przykład 3 – usługa potoku**

Najważniejsze cechy programu:

- Architektura zbliżona do architektury klient/serwer
- Działanie w domyślnej grupie
- Korzystanie z deklaratywnie utworzonego anonsu potoku (plik XML)

- Implementacja interfejsu *PipeMsgListener* w części programu usługującej jako serwer
- Implementacja interfejsu *OutputPipeListener* w części programu występującej w charakterze klienta
- Klient korzysta z usług RDV