

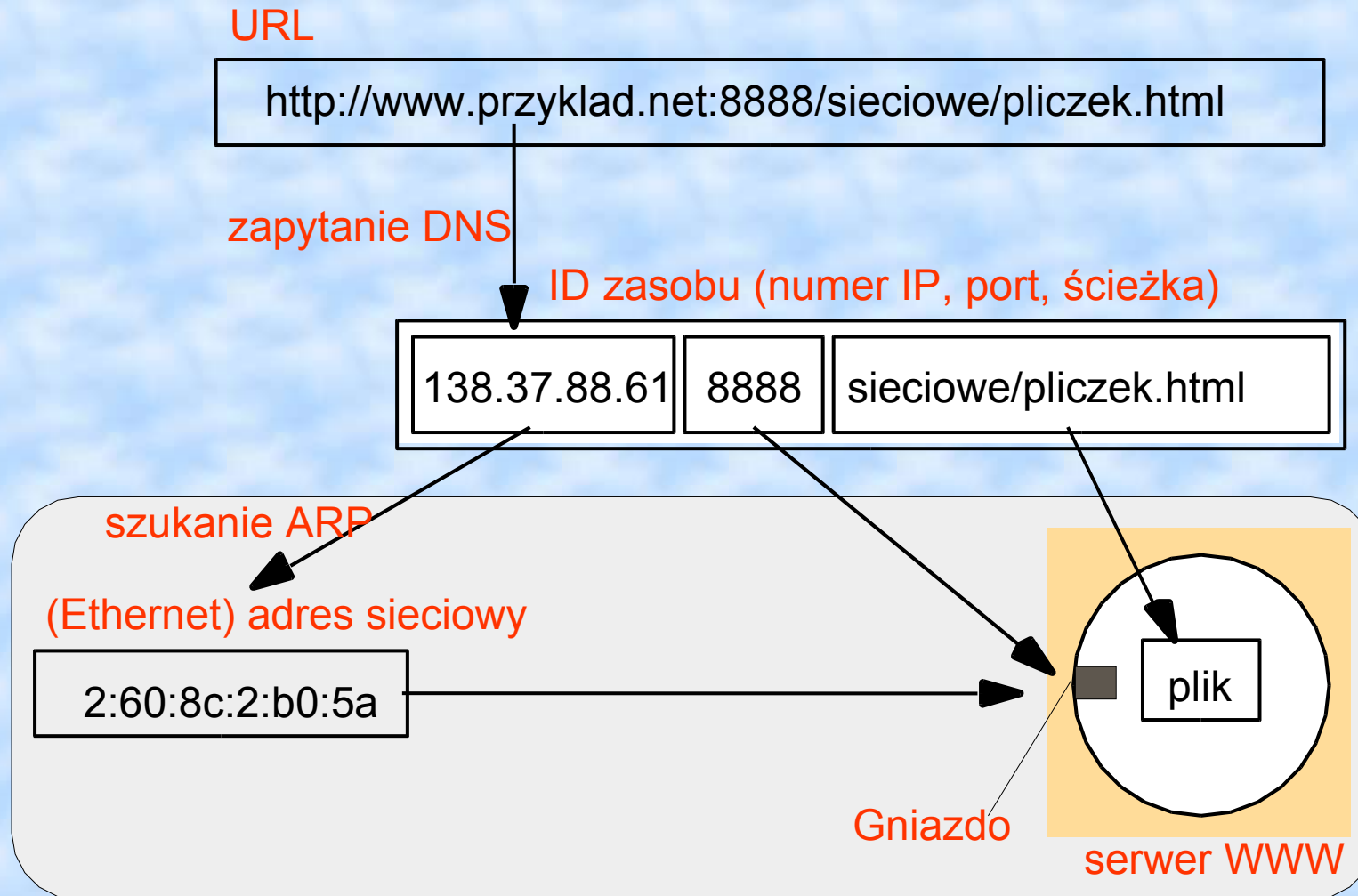
Rola nazw i usług nazewniczych

- Dostęp do zasobów jest przez referencję lub identyfikator
 - identyfikator może być przechowywany w zmiennych i może być łatwo wydobywany z tablic
 - identyfikator zawiera adres do obiektu lub może być w taki adres zamieniony
 - np. uchwyt do pliku NFS, referencja do obiektu zdalnego w CORBIE
 - nazwa jest zrozumiała dla człowieka i może być przekształcona na identyfikator bądź adres
 - nazwa serwera, nazwa pliku, numer procesu
- dla wielu celów nazwy są preferowane
 - fizyczny adres zasobu o podanej nazwie może ulec zmianie
 - mają znaczenie dla użytkowników
- zasoby można wyszukiwać po nazwach za pomocą usług nazewniczych
 - podają one identyfikatory i inne przydatne atrybuty

Wymagania dla przestrzeni nazw

- Możliwość użycia prostych ale pełnych znaczenia nazw
- Potencjalnie nieskończona liczba nazw
- Ustrukturalizowanie
 - pozwalać na podobne nazwy
 - grupować powiązane nazwy
- Pozwalać na zmianę struktury drzew nazw
 - dla pewnych klas zmian nie powinny one mieć wpływu na działanie programów
- Zarządzanie zaufaniem

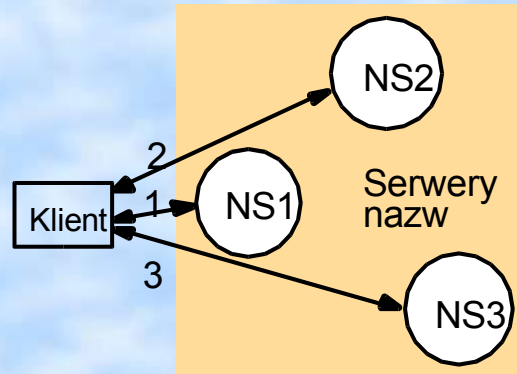
Złożone nazewnictwo domen w celu dostępu do zasobów przez URL



Nazwy i zasoby

- Aktualnie używa się różnych typów nazw dla różnych typów zasobów
 - plik – ścieżka – plik w systemie plików
 - proces – id procesu – proces na danym komputerze
 - port – numer portu – port IP na danym komputerze
- URI (Uniform Resource Identifier) pozwala na spójne indeksowanie dowolnych zasobów. Rozróżniamy dwie klasy URI:
 - URL (Uniform Resource Locator)
 - typy wyróżniane polem protokołu (http, ftp, nfs itp.)
 - część z nazwą zależy od protokołu
 - zasoby nie mogą być przemieszczane między domenami
 - URN (Uniform Resource Name)
 - wymaga usługi wyszukiwania zasobów – odpowiednika DNS-a
 - format URN:
 - urn:<przestrzeń nazw>:<nazwa w przestrzeni nazw>
 - przykład:
 - urn:ISBN:021-12547-3
 - urn:jakis_serwer.pl:NR2004-12

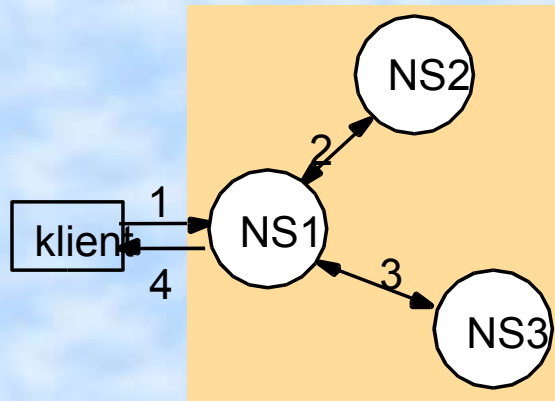
Nawigacja iteracyjna



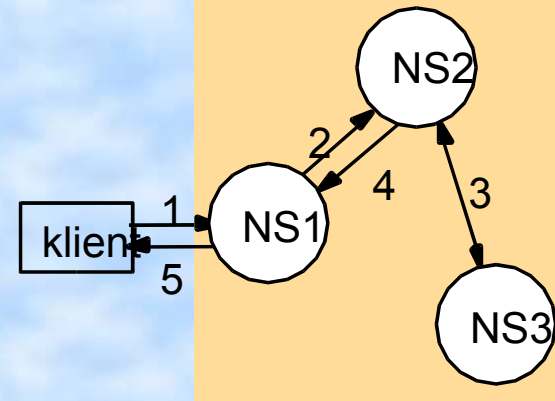
Klient iteracyjnie łączy się z serwerami nazw by przetłumaczyć nazwę

- Wykorzystywana w:
 - DNS – klient pyta o pełną nazwę rozpoczynając od lokalnego serwera. Jeśli nie zna on odpowiedzi sugeruje klientowi skontaktowanie z innym serwerem
 - NFS – klient rozdziela nazwy i prezentuje je po jednym serwerowi, razem z uchwytem do katalogu zawierającego potrzebny plik (ten sposób stosowany ze względu na możliwość wystąpienia linków symbolicznych)

Rekursywna i nie rekursywna nawigacja kontrolowana przez serwer



Nie rekursywna nawigacja



Rekursywna nawigacja

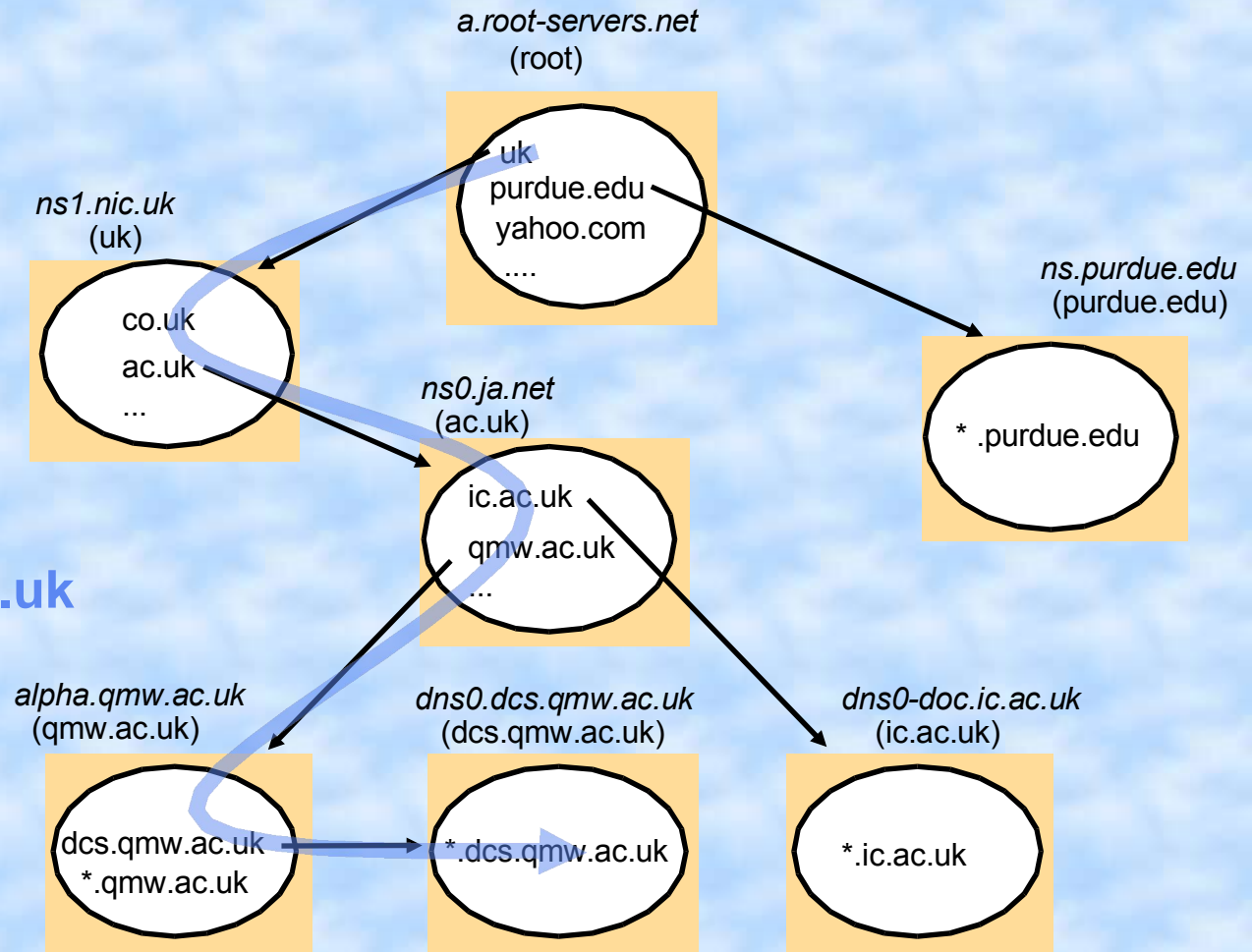
Serwer nazw komunikuje się z innymi serwerami nazw na żądanie klienta

- DNS oferuje nawigację rekursywną (opcjonalna). Zawsze musi być dostępna nawigacja iteracyjna. Nawigacja rekursywna może być wyłączana ze względów bezpieczeństwa

DNS – Domain Name System

- Rozproszona baza nazw
- Struktura nazw odzwierciedla administracyjną strukturę Internetu
- Potrafi szybko tłumaczyć nazwy na adresy IP
 - zaawansowane wykorzystywanie cache'owania
 - typowy czas odpowiedzi to około 100 ms
- Skalowalny na miliony komputerów
 - podzielona baza danych
 - cache'owanie
- Odporna na awarię serwera
 - replikacja
- Podstawowy algorytm na tłumaczenie nazw w DNS-ie
 - poszukaj nazwy w lokalnym cache'u
 - sprawdź nadrzędny serwer DNS, które odpowiedź zawiera:
 - adres innego, rekomendowanego serwera DNS
 - adres IP (niekoniecznie aktualny)

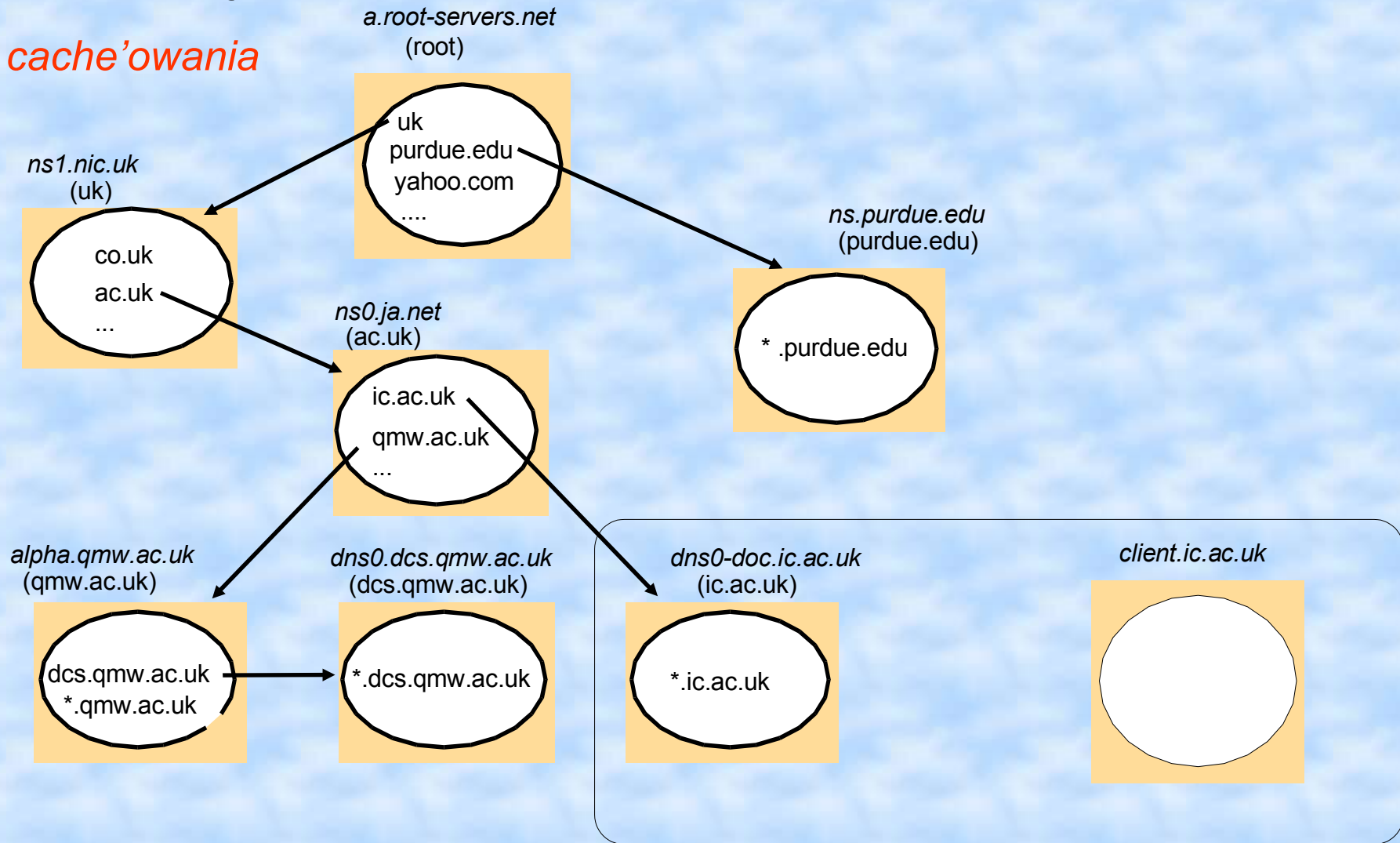
Serwery nazw DNS



Ścieżka do serwera odpowiadającego za nazwę: **jeans-pc.dcs.qmw.ac.uk**

Typowe działanie DNS-a

Bez cache'owania



Funkcje serwera DNS i konfiguracja

- Podstawowa funkcja to tłumaczenie nazw komputerów (na adresy IP)
 - cache'owanie wyników wcześniejszych zapytań (zależne od ich czasu życia)
- Pozostałe funkcje:
 - zapytania o serwer poczty dla danej domeny
 - tłumaczenie odwrotne – IP→nazwa hosta
 - informacje o hostach – typ sprzętu i OS
 - lista podstawowych usług
 - inne atrybuty...

Rekordy zasobów w DNS-ie

| <i>typ rekordu</i> | <i>znaczenie</i> | <i>zawartość</i> |
|--------------------|---|-----------------------------------|
| A | adres komputera | numer IP |
| NS | autorytatywny serwer nazw | nazwa serwera nazw |
| CNAME | nazwa kanoniczna dla aliasu | nazwa dla aliasu |
| SOA | oznacza początek danych dla strefy | parametry strefy |
| WKS | opis podstawowej usługi | lista nazw usług i protokołów |
| PTR | wskaźnika na nazwę (tłumaczenie odwrotne) | nazwa |
| HINFO | informacje o hoście | architektura maszyny i OS |
| MX | wymiana poczty | lista par <komputer, preferencje> |
| TXT | napis | dowolny tekst |

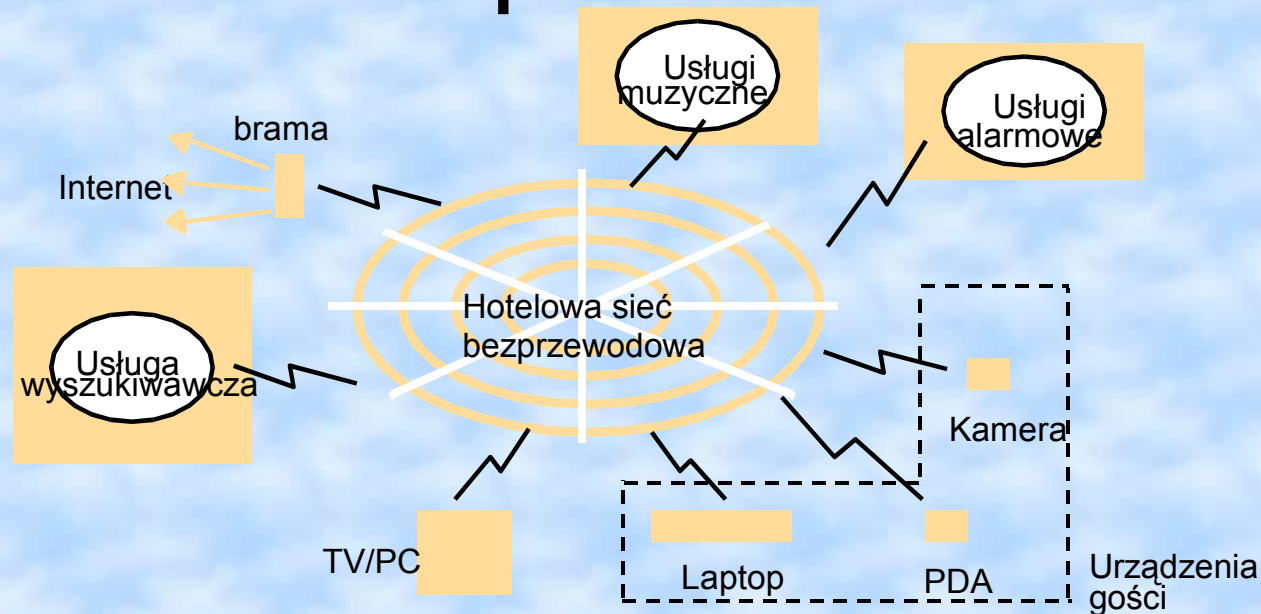
Problemy DNS

- Zmiany w tabelach nazw mogą być wolno aktualizowane ze względu na cache'owanie
 - klient odpowiada za poradzenie sobie w takich sytuacjach
- Trudno zmienić strukturę przestrzeni nazw:
 - łączenie oddzielnych domen w jeden nowy korzeń
 - przenoszenie poddrzew w inne miejsce w strukturze

Usługi katalogowe i wyszukiwawcze

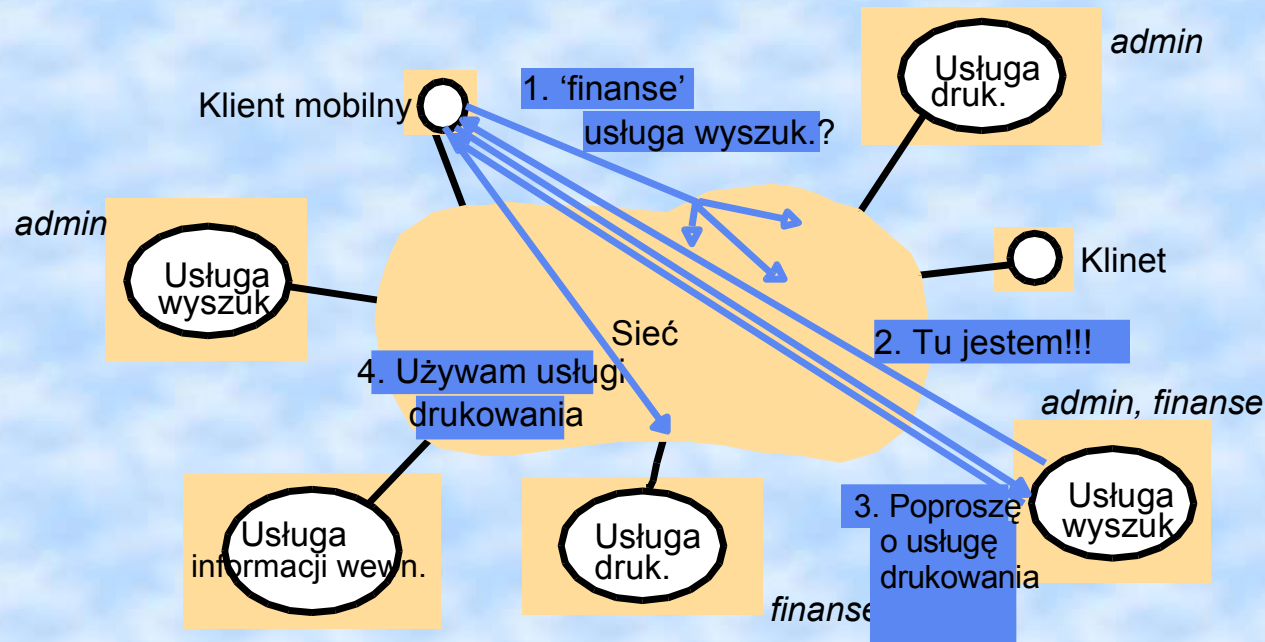
- Usługi katalogowe - „książka telefoniczna” dla zasobów w sieci
 - zwracają zbiór nazw spełniających podane kryteria
 - np. X.500, LDAP, MS Active Directory
 - (DNS przechowuje pewne dane opisowe ale są one niekompletne i DNS nie przystosowany do przeszukiwań tego typu)
- usługa wyszukiwawcza – usługa katalogowa plus:
 - jest automatycznie aktualizowana przy zmianie konfiguracji w sieci
 - odpowiada na potrzeby klientów w sieciach spontanicznych
 - wyszukuje usługi potrzebne klientowi (który może być mobilny) w aktualnym obszarze np. odszukanie najbliższej drukarki do drukowania zdjęć
 - przykłady: JINI, SLP, SSDP, SDS

Sieci spontaniczne



- Łatwa konfiguracja urządzeń gości
 - sieć bezprzewodowa
 - automatyczna konfiguracja
- Łatwa integracja z usługami lokalnymi
 - wyszukiwanie usług przydatnych gościom
- Problemy:
 - niepewne połączenia mobilne, problemy bezpieczeństwa

Wyszukiwanie usług w JINI



- Usługi JINI rejestrują swoje interfejsy i opisy w usłudze wyszukiwawczej w ich obszarze
- Klienci znajdują usługę wyszukiwawczą korzystając z multiacastu
- Usługa wyszukiwawcza JINI pozwala na szukanie wg interfejsów lub atrybutów

Czas i zegary

- Potrzebujemy dokładnie mierzyć czas:
 - by znać czas zdarzeń w komputerze
 - by to zrobić musimy zsynchronizować zegar wewnętrzny z autorytatywnym zegarem zewnętrznym
- Algorytmy synchronizacji zegarów są przydatne w:
 - kontroli współbieżności opartej na znacznikach czasowych
 - żądań uwierzytelniających np. w Kerberosie
- W systemach rozproszonych nie ma zegara globalnego
- Można używać czasu logicznego
 - pozwala określać kolejność zdarzeń
 - przydatny do kontroli spójności replikowanych danych

Zegary komputerowe i czasy zdarzeń

- Każdy komputer w rozproszonym systemie ma swój zegar wewnętrzny
 - używany przez lokalne procesy, by uzyskać aktualny czas
 - procesy na różnych komputerach mogą znakować czasem swoje zdarzenia
 - niestety zegary na różnych komputerach mogą podawać różne czasy
 - zegary komputerowe odchylają się od wzorcowego czasu i to odchylenie może być różne na różnych komputerach
 - współczynnik odchylenia zegara – rozbieżność między wskazaniami zegara komputerowego a zegara wzorcowego
- nawet jeśli zegary na wszystkich komputerach systemu rozproszonego będą ustawione identycznie, to z czasem ich wskazania będą się coraz bardziej różnić o ile nie będzie się wprowadzało odpowiednich poprawek

Zegary, zdarzenia i stany procesów

- System rozproszony definiuje się jako kolekcję P z N procesów p_i , $i=1,2,\dots,N$
- Każdy proces p_i ma stan s_i , który tworzy zawartość jego zmiennych
- Procesy komunikują się wyłącznie przy pomocy komunikatów (przez sieć)
- Akcje wykonywane przez procesy: wyślij, odbierz, zmień swój stan
- Zdarzenie: wykonanie jednej akcji przez w w trakcie działania procesu
- Zdarzenia wywołane przez jeden proces p_i są w relacji porządku liniowym między sobą
 - $e \rightarrow_i e'$ wtedy i tylko wtedy gdy wydarzenie w było przed wydarzeniem e' w procesie p_i
- Historię procesu p_i tworzy sekwencja wydarzeń $historia(p_i) = h_i = \langle e_{i0}, e_{i1}, e_{i2}, \dots \rangle$

Zegary

- Wiemy jak szeregować wydarzenia
- Do znakowania czasowego używamy zegara komputera
- W czasie rzeczywistym OS odczytuje czas ze sprzętowego zegara komputera
- Wylicza czas systemowy
 - np. jako liczbę mikrosekund od pewnego czasu w historii
 - w ogólności wskazania zegara nie są idealnie dokładne
 - jeśli zegar zachowuje się wystarczająco dobrze może być użyty do znakowania zdarzeń wewnątrzprocesowych

Odchylenia zegarów w systemie rozproszonym



Sieć

- Zegary komputerowe w ogólności nie pokazują dokładnie tych samych czasów
- Odchylenie (skew) – różnica między wskazaniem dwóch zegarów
- Zegary komputerowe są skłonne do odchylenia (mierzą czas z różną częstotliwością)
- Współczynnik odchylenia zegara – różnica wskazań zegara względem zegara wzorcowego przypadająca na jednostkę czasu
- Typowe zegary mają odchylenie około 1 s. na 11-12 dni (10^{-6} s/s)
- Wysokiej jakości zegary mają wsp. odchylenia równy ok. 10^{-7} s/s

UTC – Uniwersalny Czas Skoordynowany

- Międzynarodowy czas atomowy jest mierzony przez bardzo dokładne zegary (wsp. odchylenia wskazań 10^{-13})
- UTC jest międzynarodowym standardem przechowywania czasu
- Jest ustalany na podstawie wskazań zegarów atomowych i od czasu do czasu jest synchronizowany z czasem astronomicznym
- Jest nadawany kanałami radiowymi (np. GPS)
- Komputery posiadające odpowiednie odbiorniki mogą synchronizować się z tymi kanałami
- Sygnały stacji naziemnych mają dokładność 0.1-10 ms
- Sygnały z GPS mają dokładność ok. 1 μ s

Synchronizowanie zegarów fizycznych

- Synchronizacja zewnętrzna:
 - zegar komputera C jest synchronizowany z zewnętrznym źródłem autorytatywnym S w taki sposób, by:
 - $|S(t)-C(t)| < D$ przez pewien określony czas I
 - zegar C jest dokładny w granicach określonych przez D
- Wewnętrzna synchronizacja:
 - dwa komputery synchronizują się nawzajem tak by:
 - $|C1(t)-C2(t)| < D$ przez pewien określony czas I
 - zegary C1 i C2 są zgodne w granicach określonych przez D
- Wewnętrznie synchronizowane zegary mogą być niesynchronizowane z zegarem wzorcowym
- Jeśli zbiór procesów P jest zsynchronizowany zewnętrznie w granicach określonych przez D to wewnętrznie są zsynchronizowane w granicach określonych przez 2D

Poprawność zegara

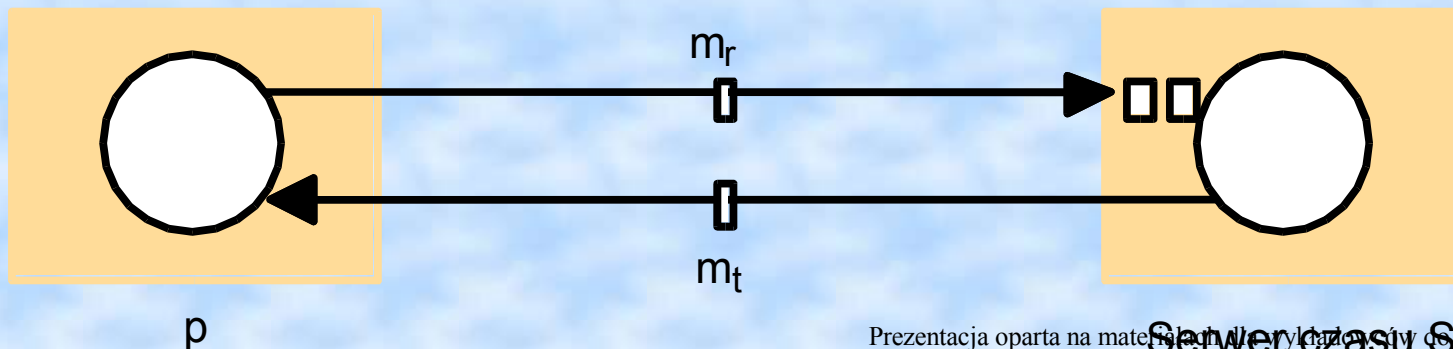
- Zegar sprzętowy H jest poprawny jeśli jego współczynnik odchylenia zegar mieści się w granicach $\rho > 0$ (np. 10^{-6} s/s)
- Oznacza to, że błąd w pomiarach okresu czasu rzeczywistego między t a t' jest ograniczony przez:
 - $(1 - \rho) (t' - t) \leq H(t') - H(t) \leq (1 + \rho) (t' - t)$ (gdzie $t' > t$)
- Słaby warunek monotoniczności:
 - $t' > t \Rightarrow C(t') > C(t)$
 - takie wymagania stawia wiele programów, np. *make*
- Wadliwy zegar to taki, który nie spełnia warunku poprawności
- *crash failure* – zegar się zatrzymuje
- *arbitralne błędy* – inne błędy zegara, np. przeskoki w czasie

Synchronizacja w systemie synchronicznym

- Synchroniczny system rozproszony to taki, który spełnia poniższe warunki:
 - czas wykonania każdego kroku procesu ma znane dolne i górne ograniczenia czasowe
 - każda wysłana wiadomość musi być odebrana w pewnych granicach czasowych
 - każdy proces ma swój zegar logiczny, którego wsp. odchylenia od czasu rzeczywistego ma znane granice
- Wewnętrzna synchronizacja w systemie synchronicznym
 - proces p1 wysyła swój czas lokalny t do procesu p2 w wiadomości m
 - p2 może ustawić swój zegar na $t + T_{trans}$ gdzie T_{trans} jest czasem transmisji m
 - T_{trans} jest nieznanym ale znamy $min \leq T_{trans} \leq max$
 - $u = max - min$. Ustawiamy zegar na $t + u/2$ wtedy odchylenie jest mniejsze od $u/2$

Algorytm Cristiana dla systemów asynchronicznych

- Serwer czasu S otrzymuje sygnał ze źródła UTC
 - proces p pyta o czas w momencie m_r i otrzymuje t w momencie m_t od S
 - p ustawia czas zegar na $t + T_{\text{tam_i_z_powrotem}} / 2$
 - dokładność ustawienia to $\pm (T_{\text{tam_i_z_powrotem}} / 2 - \text{min})$ (min to minimalny czas wymiany tych komunikatów):
 - najwcześniejszy czas jaki może wysłać S to min od wysłania zapytania o czas
 - najpóźniejszy czas to min przed przybyciem odpowiedzi do p
 - czas na zegarze S w momencie przybycia odpowiedzi do t jest w przedziale $[t + \text{min}, t + T_{\text{tam_i_z_powrotem}} - \text{min}]$



Algorytm z Berkeley

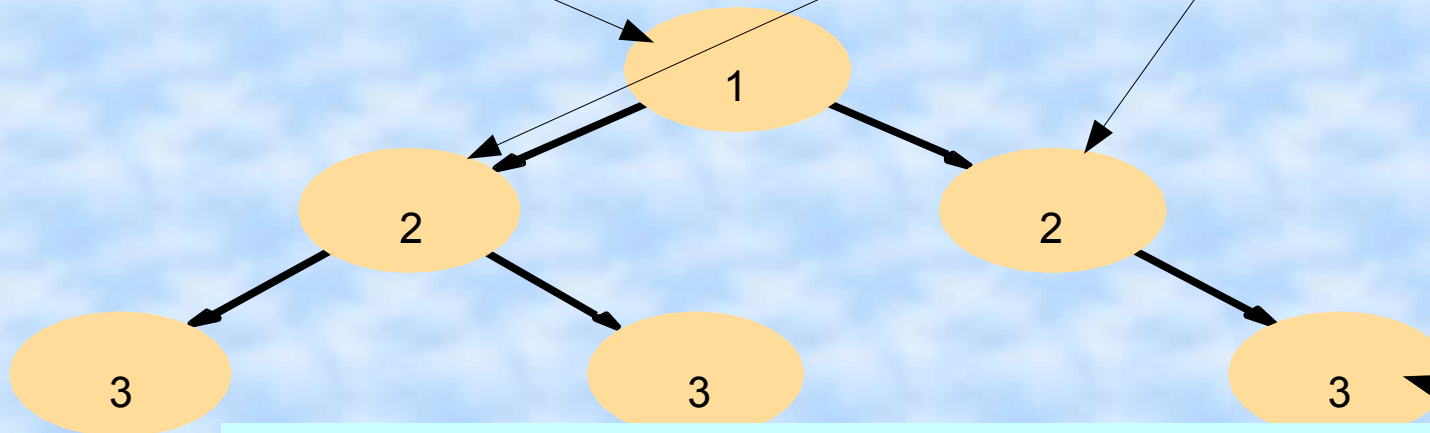
- Algorytm Cristiana
 - pojedynczy serwer czasu może się zepsuć, więc sugeruje się użycie grupy serwerów
 - nie uwzględnia wadliwie działających serwerów
- Algorytm z Berkeley
 - algorytm wewnętrznej synchronizacji grupy komputerów
 - master odpytuje czasy innych serwerów (slave)
 - master używa czas wymiany komunikatów do ustalenia czasów na serwerach podrzędnych
 - następnie brana jest średnia otrzymanych czasów (z wyrzuceniem bardzo złych (podejrzanych) czasów)
 - komputerom podrzędnym wysyła informacje o korekcie czasu
 - testy działania algorytmu:
 - dla 15 komputerów czas synchronizacji to 20-25 ms, wsp. odch. $<2 \times 10^{-5}$
 - jeśli master się zepsuje można zastosować algorytm wyboru nowego

NTP – Network Time Protocol

- Usługa czasowa dla internetu – pozwala na synchronizację klientów z UTC
- Wiarygodność uzyskiwana dzięki wielu ścieżkom, skalowalny, umożliwia uwierzytelnianie źródeł czasu

Serwery główne są podłączone do źródeł czasu UTC

Serwery drugorzędne synchronizują się z serwerami głównymi



Podsieć synchronizacji

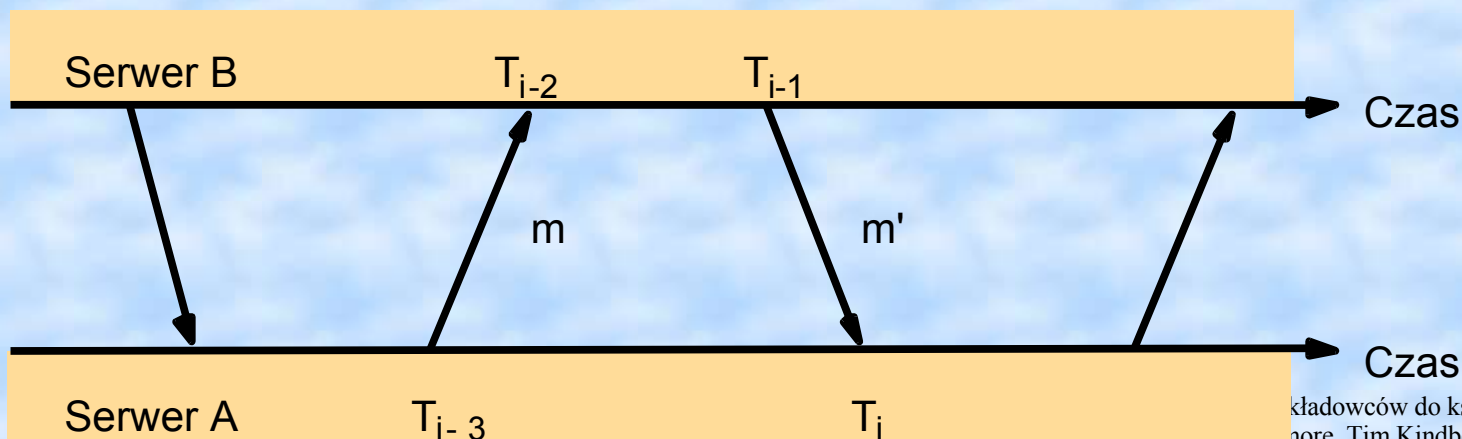
serwery najniższego poziomu są w komputerach użytkowników

NTP – synchronizacja serwerów

- Podsieć synchronizacji może się zrekonfigurować jeśli:
 - serwer główny utraci kontakt ze źródłem UTC stając się serwerem drugiego poziomu
 - serwer drugiego poziomu, który straci swój serwer główny może zmienić poszukać inny serwer główny
- Tryby synchronizacji:
 - multicast
 - w przypadku szybkie łącza LAN serwer wysyła multicastem wskazania, a klienci uwzględniają pewien niewielki czas opóźnienia (niezbyt dokładne)
 - wywołanie procedury
 - serwer akceptuje zapytania (używany np. algorytm Cristiana). Większa dokładność. Nie zawsze można użyć multICASTu.
 - symetryczna
 - para serwerów wymienia wiadomości ze wskazaniem czasu
 - używany gdy potrzebna duża dokładność synchronizacji

Wymiana wiadomości między równorzędnymi serwerami NTP

- Wszystkie tryby używają UDP
- Każda wiadomość zawiera znaczniki czasowe ostatnich wydarzeń:
 - czasy lokalne wysłania i odbioru ostatnich wiadomości
 - czas lokalny wysłania aktualnej wiadomości
- Odbiorca zapamiętuje czas otrzymania $T(i)$ (mamy $T(i-3)$, $T(i-2)$, $T(i-1)$, $T(i)$)
- W trybie symetrycznym może wystąpić nie pomijalne opóźnienie między wiadomościami



Dokładność NTP

- Dla każdej pary wiadomości wymienianej między serwerami NTP oblicza odległość $o(i)$ pomiędzy wskazaniami obydwu zegarów i opóźnienie $d(i)$ (łącznie czas wiadomości z czasami t i t')

$$T_{i-2} = T_{i-3} + t + o \text{ and } T_i = T_{i-1} + t' - o$$

- Daje to nam:

$$d_i = t + t' = T_{i-2} - T_{i-3} + T_i - T_{i-1}$$

- oraz:

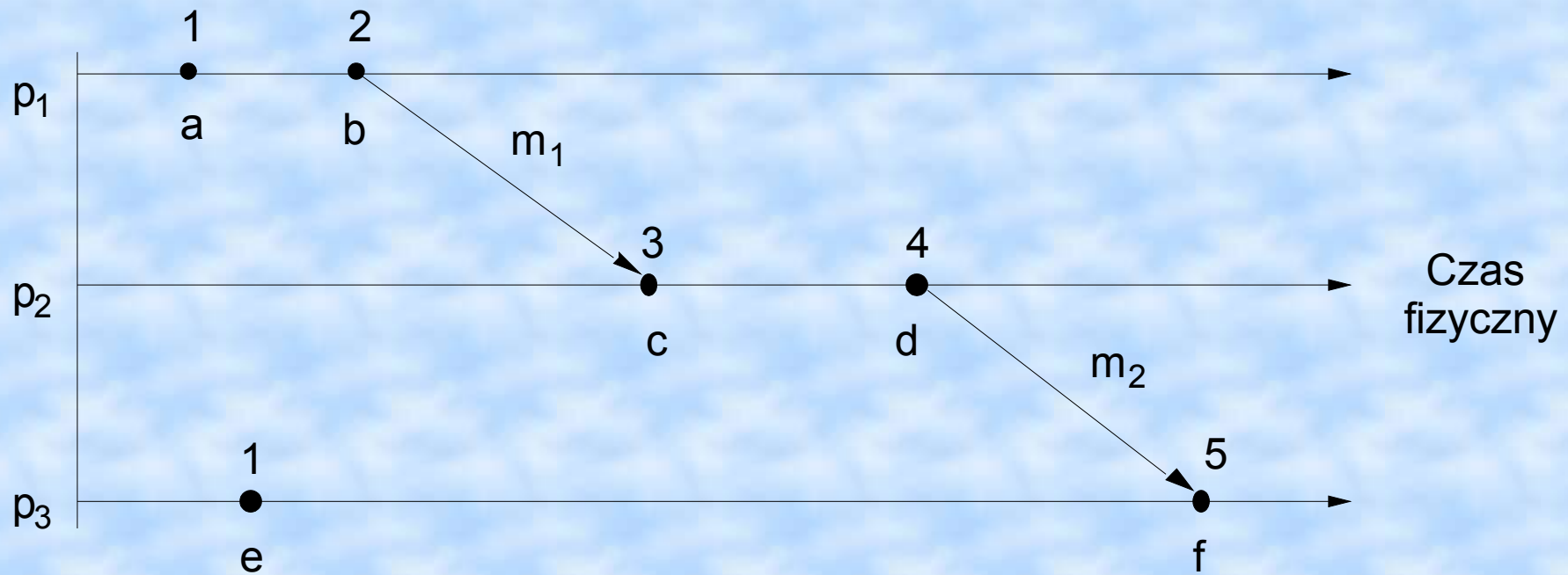
$$o = o_i + (t' - t)/2 \text{ where } o_i = (T_{i-2} - T_{i-3} + T_i - T_{i-1})/2$$

- Korzystając z wiedzy, że $t, t' > 0$ można wykazać, że:

$$o_i - d_i/2 \leq o \leq o_i + d_i/2 .$$

- stąd $o(i)$ jest oszacowaniem odległości a $d(i)$ jest miarą dokładności tego oszacowania
- NTP używają $o(i)$ i $d(i)$ do oceny wiarygodności komunikatów i wyboru partnerów do synchronizacji
- Dokładność jaką można uzyskać korzystając z NTP to ok. 30 ms w Internecie i 1 ms w przypadku LAN

Zegary Lamporta



Zegary wektorowe

