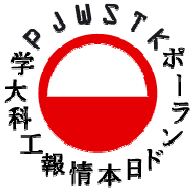


Polsko-Japońska Wyższa Szkoła Technik Komputerowych

WS-Transaction (WS-TX) i transakcje z kontrolowaną przejrzystością

Poza ACID i 2PC – koncepcje mechanizmów transakcji wspierających procesy biznesowe

Edgar Głowacki
edgar@glowacki.eu

ACID i 2PC – słowa klucze

- ACID i 2PC – słowa klucze nieodmiennie kojarzone z transakcjami
  - ✓ wpływają na rozumienie mechanizmu transakcji
- ACID – bardzo restrykcyjny paradygmat
  - ✓ zastosowanie przy bardzo prostych transakcjach – np. bankowych – do tej pory najczęściej pojawiający się przykład ilustrujący zasadę działania
- 2PC – przeniesienie założeń ACID do przetwarzania rozproszonego

Transakcje w życiu codziennym (1)

- W rozumieniu potocznym transakcja to wymiana dóbr i usług
- Transakcje znane z życia nie wpisują się w model ACID
  - ✓ ludzie nie wykazują zrozumienia dla konieczności powtórzenia wykonanych już czynności – np. konieczność ponownej obsługi wniosku od początku (złożenie, zaopiniowanie, …), w sytuacji, gdy na pewnym etapie jego przetwarzanie się nie powiodło z przyczyn drugorzędnych – np. chwilowa niedostępność usługi danego urzędu
  - ✓ wykonane już operacje mogą wiązać się z wymiernymi kosztami, których nie można cofnąć – np. wykonanie ekspertyzy zewnętrznej
  - ✓ może istnieć konieczność wycofania części operacji po zatwierdzeniu transakcji – np. w wyniku powstania błędu – zarówno po stronie ludzkiej, jak i automatu usprawniającego realizację danej operacji

Transakcje w życiu codziennym (2)

- Transakcje znane z życia nie wpisują się w model ACID (c.d.)
  - ✓ mogą się powieść częściowo – np. wniosek przeszedł pomyślnie jakąś procedurę weryfikacji, ale dalszych kroków pierwotnie założonej procedury nie można wykonać – można zamknąć transakcję nie cofając dotychczasowych efektów i przetwarzać wniosek zgodnie z inną procedurą
  - ✓ mogą przeciągać się w czasie – czasami wręcz w nieskończoność
  - ✓ zazwyczaj są nieprzezroczyste dla innych transakcji – mają skutki uboczne widoczne na zewnątrz – np. wykonanie płatności kartą kredytową przy zakupie przez internet
  - ✓ niektórzy uczestnicy transakcji mogą się wycofać, co nie oznacza zerwania transakcji – może nie udać nam się wynająć samochodu, ale to nie znaczy, że zrezygnujemy z wakacji – jesteśmy skłonni „zaryzykować” i spróbować zorganizować sobie środek transportu już na miejscu

ACID, 2PC i akcje kompensujące

- ACID i 2PC zakładają
  - ✓ przebieg zerojedynkowy
  - ✓ przezroczystość
  - ✓ całkowitą odwracalność
  - ✓ brak zbyt dużych opóźnień
- Akcje kompensujące
  - ✓ odwracają skutki uboczne – np. wysyłając mail odwołujący poprzedni
  - ✓ wszystkie pozostałe ograniczenia 2PC pozostają

5

WS-Coordination – aktywność

- Aktywność – „jednostka obliczeniowa” (*computational units*) utworzona w wyniku powiązania ze sobą wielu uczestników za pomocą WS
- Zazwyczaj złożona – niejednokrotnie tworząca skomplikowane relacje pomiędzy uczestnikami
- Realizacja aktywności może trwać długo
  - ✓ opóźnienia wpisane w charakter wspieranej działalności operacyjnej
  - ✓ wymagana interakcji z użytkownikiem

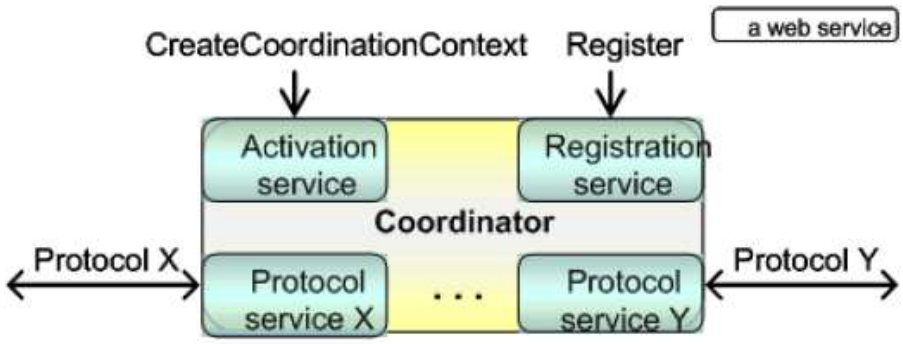
6

WS-Coordination – fundament szkieletu WS-TX

- Rozszerzalny szkielet umożliwiający koordynację aktywności
  - ✓ szkielet umożliwia uzgodnienie uczestnikom wyniku koordynowanej aktywności – wynik nie jest zerojedynkowy
  - ✓ koordynator – usługa zapewniająca koordynację działań uczestników
  - ✓ protokoły koordynujące dla różnych rodzajów aktywności – definiowane przez specyfikacje bazujące na WS-Coordination
  - ✓ protokoły koordynujące mogą dotyczyć różnego rodzaju aktywności zarówno krótkotrwałych (s*hort-lived*), jak i długotrwałych (*long-lived*)

7

WS-Coordination – koordynator aktywności (1)



8

WS-Coordination – koordynator aktywności (2)

- Funkcjonalność koordynatora (*CoordinationService*) obejmuje
  - ✓ *Activation Service* – usługę umożliwiającą tworzenie kontekstu koordynującego aktywność
  - ✓ *Registration Service* – udostępnia operację *Register* umożliwiającej WS dołączenie do aktywności
  - ✓ dodatkowe usługi określone przez dany protokół koordynacji budowany w oparciu o WS-Coordination

9

WS-Coordination – *CoordinationContext* (1)

- Kontekst koordynacji
  - umożliwia wymianę informacji dającej podstawę do koordynacji współdziałania stron biorących udział w aktywności
  - jest dystrybuowany wśród uczestników za pośrednictwem mechanizmów określonych przez daną aplikację – np. w nagłówkach SOAP
  - umożliwia rejestrację w ramach aktywności
  - określa rodzaj koordynacji
  - może zawierać rozszerzenia specyficzne dla danego protokołu – jak w przypadku WS-BusinessActivity

10

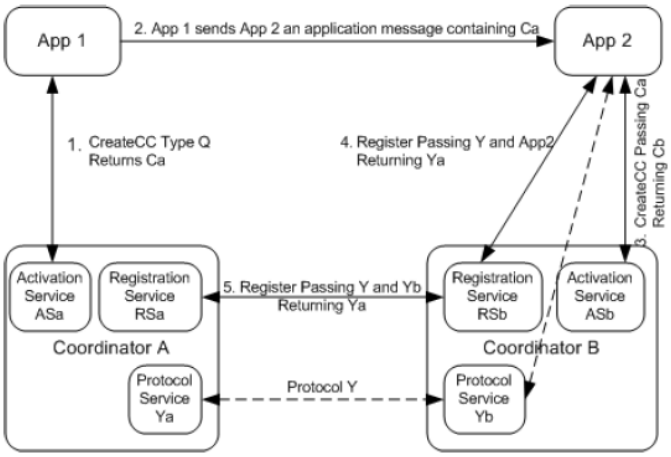
WS-Coordination – *CoordinationContext* (2)

- Kontekst koordynacji (c.d.)
  - po pobraniu kontekstu aplikacja przekazuje go innym uczestnikom aktywności – w sposób określony przez protokół pochodny
  - WS po uzyskaniu kontekstu rejestruje się do aktywności za pomocą *Registration Service*, która może być wskazana w kontekście bądź też przez zaufanego koordynatora
  - przepływ kontekstu (*context flow*) – przekazywanie kontekstu między uczestnikami (*parties*)

WS-Coordination – rozszerzenia

- Rozszerzenia WS-Coordination obejmują
  - ✓ budowę protokołów koordynujących – WS-Coordination nie określa żadnego
  - ✓ wybór protokołu koordynującego na podstawie rodzaju aktywności
  - ✓ rozszerzenia struktur wprowadzonych w schemacie WS-Coordination
  - ✓ przykładowe rozszerzenie – wymiana informacji o poziomie izolacji transakcji realizowanej w ramach aktywności

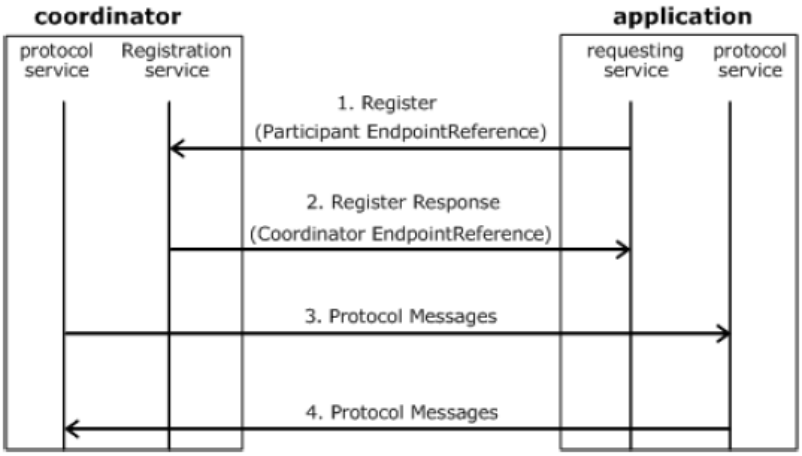
WS-Coordination – przykładowy scenariusz (1)



WS-Coordination – przykładowy scenariusz (2)

1. WS1 wysyła żądanie do koordynatora A o utworzenie kontekstu dla koordynacji typu Q
2. W wyniku uzyskuje kontekst Ca od koordynatora, kontekst zawiera *endpoint reference* do usługi RS koordynatora
3. WS1 przekazuje kontekst w komunikacie zaadresowanym do WS2
4. WS2 preferuje koordynatora B, któremu przekazuje otrzymany kontekst – w wyniku tego koordynator B tworzy kontekst Cb zawierający ten sam identyfikator aktywności co Ca
5. WS2 ustala rodzaje protokołów wykorzystywanych w typie koordynacji Q i wybiera protokół Y
6. Koordynator B przekazuje informację o wyborze WS2 do Koordynatora A wykorzystując jego usługę RS – w żądaniu rejestracji jest endpoint reference do implementacji protokołu Y po stronie koordynatora A
7. Koordynator A przesyła endpoint reference do własnej implementacji Y
8. Od tej pory koordynatorzy A i B komunikują się za pomocą Y w ramach danej aktywności

WS-Coordination – rejestracja uczestnictwa



Aplikacją rejestrującą może być koordynator działający w imieniu WS – jak w podanym przykładzie

15

WS-BusinessActivity

- WS-BusinessActivity – rozszerzenie WS-Coordination
  - ✓ wprowadza kontekst aktywności biznesowej – rodzaj kontekstu koordynacyjnego zdefiniowanego w WS-Coordination
  - ✓ wszystkie komunikaty wykorzystujące WS-BusinessActivity muszą korzystać z kontekstu aktywności biznesowej
- Dodatkowe ograniczenia
  - ✓ koordynator WS, która otrzyma kontekst staje się zawsze podrzędny względem koordynatora WS, od której pochodzi komunikat zawierający kontekst
  - ✓ jeśli w nagłówku żądania bloku WS-BusinessActivity nie ma kontekstu aktywności biznesowej, koordynator odbiorcy musi go utworzyć

16

WS-BusinessActivity – kontekst

- Kontekst aktywności biznesowej
  - ✓ może zawierać informację o czasie ważności (element *Expires*) – okres czasu odliczany od momentu utworzenia/otrzymania kontekstu
  - ✓ po wygaśnięciu ważności kontekstu aktywność może zostać zakończona
- W chwili wygaśnięcia kontekstu
  - ✓ koordynator może podjąć decyzję o odwołaniu aktywności, bądź też wykonaniu operacji kompensującej
  - ✓ uczestnik aktywności może zdecydować się ją opuścić

17

WS-BusinessActivity – typy koordynacji

- *AtomicOutcome* – koordynator tego rodzaju koordynacji wymusza albo *Close* albo *Compensate* na wszystkich uczestnikach
  - ✓ jest to podstawowy rodzaj wspierany przez wszystkie implementacje
- *MixedOutcome* – wynik poszczególnych uczestników może być różny – *Close* albo *Compensate*

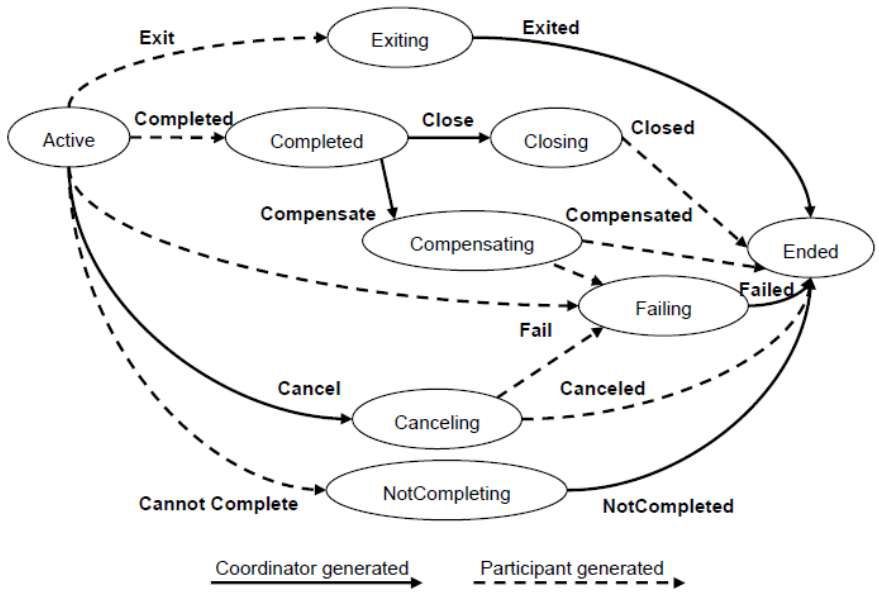
18

WS-BusinessActivity – rodzaje protokołów

- *BusinessAgreementWithParticipantCompletion*
  - ✓ uczestnik rejestruje się w koordynacji z własnym koordynatorem, ale sam wie, kiedy zakończył wszystkie operacje związane z daną aktywnością
- *BusinessAgreementWithCoordinatorCompletion*
  - ✓ uczestnik rejestruje się w koordynacji z własnym koordynatorem, który decyduje, czy uczestnik wykonał wszystkie operacje w ramach koordynacji

19

WS-BusinessActivity – …*ParticipantCompetion* (1)



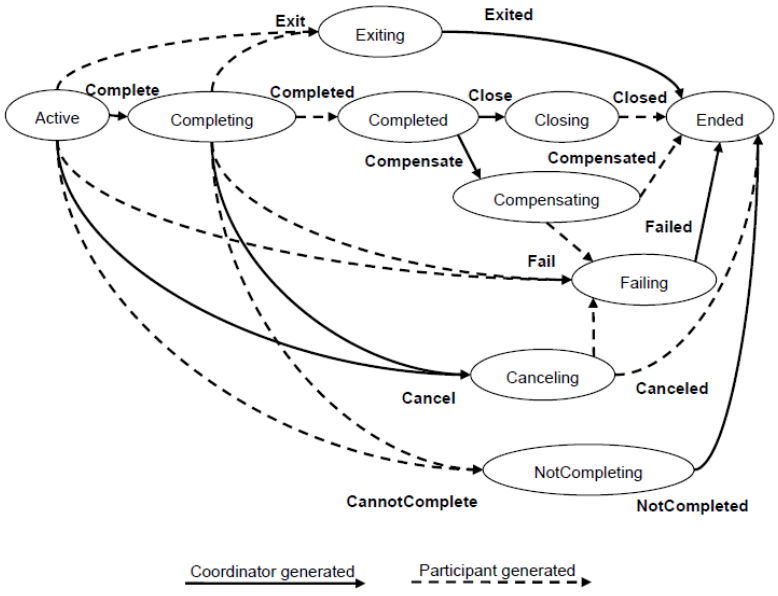
20

WS-BusinessActivity – …*ParticipantCompetion* (2)

- *BusinessAgreementWithParticipantCompletion*
  - ✓ Komunikaty odbierane przez koordynatora
    - ❖ *Completed, Fail, Compensated, Closed, Canceled, Exit, CannotComplete*
  - ✓ Komunikaty odbierane przez uczestnika
    - ❖ *Close, Cancel, Compensate, Failed, Exited, NotCompleted*
  - ✓ Komunikaty odbierane przez obie strony
    - ❖ *GetStatus, Status*

21

WS-BusinessActivity – …*CoordinatorCompletion* (1)



22

WS-BusinessActivity – …*CoordinatorCompletion* (2)

- *BusinessAgreementWithCoordinatorCompletion* – różnice w stosunku do …*ParticipantCompetion*
  - ✓ Komunikaty odbierane przez koordynatora
    - ❖ *Fail*, *CannotComplete*
  - ✓ Komunikaty odbierane przez uczestnika
    - ❖ *Complete*

23

Transakcje z kontrolowaną przejrzystością (1)

- Pomysł wywodzący się z dziedziny baz danych
- Transakcja – koncepcyjnie zbliżona do kontekstu WS-Coordination
  - ✓ jest specyficzną procedurą znaną z języków imperatywnych
  - ✓ nie jest bytem niewidocznym dla programisty – jest obiektem, którego właściwości można w każdej chwili odpytać
  - ✓ obiekt transakcji posiada właściwości (nazwa, status, …), które mogą być odpytywane
  - ✓ metody transakcji mogą zwracać listę, umożliwiać przerwanie, …
- Logi transakcji
  - ✓ obiekty z możliwością odpytywania – podobnie jak sama transakcja
  - ✓ metody logu powinny m.in. umożliwiać wyszukanie miejsc, które zostały zmodyfikowane przez transakcję, …

24

Transakcje z kontrolowaną przejrzystością (2)

- Hierarchia transakcji – 2 tryby
  - ✓ tryb zwykły (tradycyjny) – transakcja zagnieżdżona zatwierdza zmiany i przekazuje zamki do transakcji nadrzędnej
  - ✓ tryb oczekiwania na zatwierdzenie (*wait-for-commit*) – zasadniczo 2PC – transakcja podrzędna kończy przetwarzanie i czeka na sygnał o zatwierdzeniu pochodzący od transakcji nadrzędnej – umożliwia cały czas kontrolę nad użytymi zasobami i w razie potrzeby wykonanie akcji kompensującej
- Monitor transakcji
  - ✓ zasadniczo tradycyjny
  - ✓ dopuszcza różne poziomy izolację – z lokalnymi kopiami stron – w rodzaju ADO.NET z koniecznością rozwiązywania konfliktów

25

Transakcje z kontrolowaną przejrzystością (3)

- Wycofywanie zatwierdzonych transakcji
  - ✓ oparte na logu transakcji rejestrującym historię zmian – coś w rodzaju systemu wersjonowania
  - ✓ nie ma problemu – poza ew. skutkami ubocznymi – kiedy transakcja nie została jeszcze nadpisana
  - ✓ jeśli została nadpisana decyzja o tym, w jaki sposób przywrócić stan spójny powinna w jakimś stopniu zależeć od człowieka (transakcje prezentowane w postaci drzewa)
  - ✓ może okazać się trudne doprowadzenie do spójności bez ponownego wykonania transakcji nadpisujących poprzedzonym wykonaniem operacji kompensującej – sam zapis zmian typu kontrola wersji nie uwzględni bowiem reguł biznesowych – chyba, że przy każdej zmianie jest określona odpowiednia reguła, co stanowczo zwiększy stopień skomplikowania rozwiązania

26

Transakcje z kontrolowaną przejrzystością (4)

- Globalne zatwierdzenie transakcji przy wycofaniu transakcji lokalnych
  - ✓ koncepcyjnie zbliżone do trybu *MixedOutput* WS-BusinessActivity

27