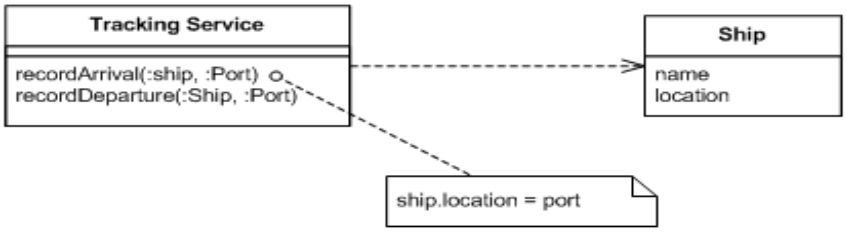
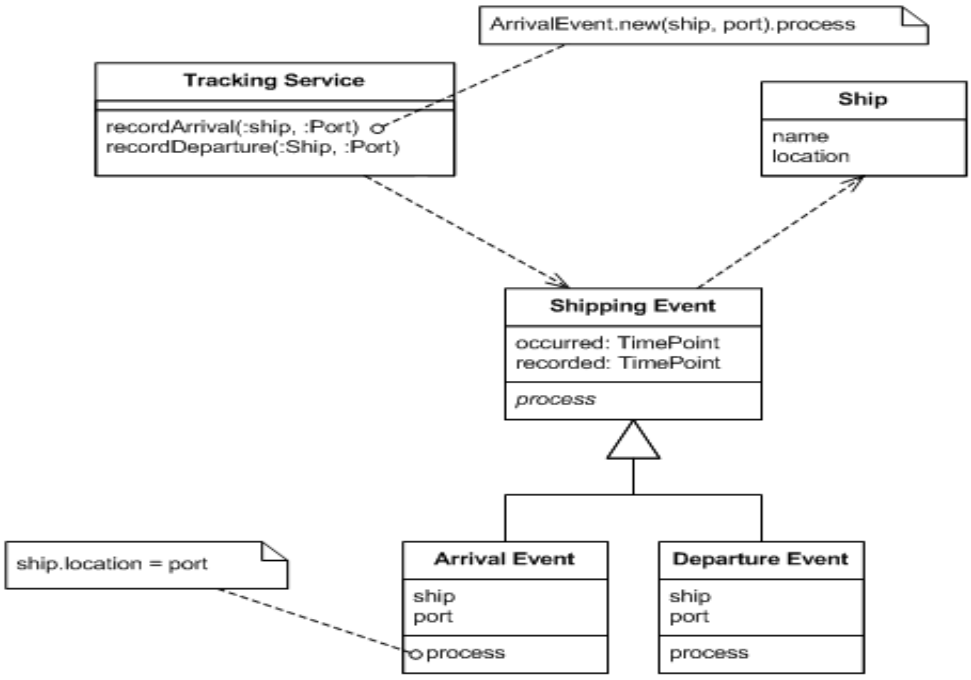
Event Sourcing

Modelul asigură că toate schimbările aduse stării unei aplicații sunt stocate ca o succesiune de evenimente într-o “listă de evenimente” , se stochează toate schimbările sistemului , nu numai starea sa curentă este salvată. Aplicația poate reconstrui starea curentă ,prin reînlocuirea de evenimente trecute , aplicația va selecta cel mai recent (chiar instantaneu) eveniment care a avut loc.

Ideea fundamentală a Event Sourcingului este aceea de a asigura că orice schimbare a stării unei aplicații este capturată într-un obiect de evenimente și că aceste obiecte de evenimente sunt stocatate în ordinea în care au fost aplicate la un moment dat. Un exemplu al utilizării modelului ar fi că ,avem multe nave într-o mare , pentru a urmări fiecare navă unde este, am avea nevoie de o aplicație de urmărire cu metode care să ne permită să aflam când o navă ajunge sau pleacă de la un port la altul.



Atunci când serviciul este apelat ,acesta găsește nava căutată și actualizează locația. Obiectele navei înregistreaza starea curentă a navelor. Introducând un Event Sourcing ,acesta creează un obiect de tipul eveniment care va înregistra modificările , si va face actualizările navelor pe baza acestor modificări.



Sa ne imaginăm că sunt 3 nave , Nava1 , Nava2 ,Nava3 , ele pleacă în , San Diego , Marseille, Constanța . Cu un serviciu standard am putea observa doar destinația finală ale navelor ,dar cu Event Sourcing putem stoca permanent evenimentele , ele vor fi persistente la fel ca obiectele navei , astfel noi putem vedea starea curentă a navei (folosindun-ne de înregistrarea de evenimente , reparcurgând traseul) sau putem vizualiza istoricul navei , prin ce porturi a mai trecut nava . Cheia acestui modul este ca toate modificările aduse obiectelor de tip domeniu sunt inițiate de obiectele de tip eveniment ,acest lucru conduce la o serie de facilități care pot fi construite în jurnalul de evenimente .

Principalele beneficii:

1.Reconstructia completă ,adică putem elimina de tot starea curentă a și să o reconstruim doar cu evenimente din jurnalul de evenimente sau ne putem întoarce la un anumit eveniment pentru a vedea în ce stare a fost aplicația la un momendat , vom avea un istoric total al modificărilor pe care aplicația le-a avut de a lungul timpului.

2.Interogarea temporară , adică putem determina starea aplicației în orice moment deoarece apariția evenimentului menține istoricul complet al fiecărui obiect de activitate , implementarea interogărilor temporale și reconstituirea stării istorice a unei entități este simplă.

3.Repetarea evenimentului , de exemplu , constatăm ca avem o eroare in aplicație , ne putem întoarce la un eveniment trecut, o stare mai veche a aplicației , care a fost implementată greșit , calculăm consecințele schimbării acelui eveniment si a evenimentelor ulterioare ,stările viitoare ale aplicației , care vor fi modificate implicit de o schimbare a acelui eveniment si vedem dacă schimbarea e posibilă. De exemplu aplicația a trecut prin 10 stări unde am aflat că avem o eroare , folosim istoricul si vedem că în starea a 8 este o eroare si ne întoarcem să o modifcăm , dar și stările 9 si 10 vor fi modificate.

4. Rezolvă una dintre problemele cheie în implementarea unei arhitecturi bazate pe evenimente și face posibilă publicarea fiabilă a evenimentelor ori de câte ori se schimbă starea aplicației.

Modele similare ale Event Sourcing ar fi :

-Saga și Domain event pattern.

- CQRS adesea utlizat împreună cu Event Sourcing.

- Audit Loggin pattern.