Caragea Sebastian-Dacian,
Ingineria Sistemelor, anul 4, grupa 1

Domain driven design – domain events

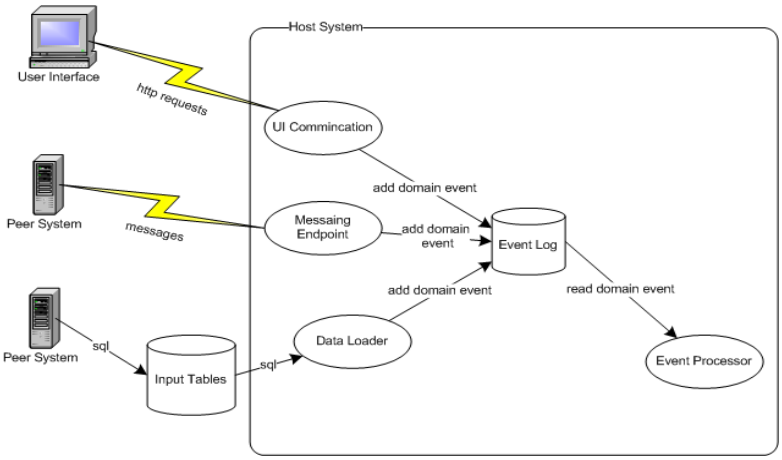
Domain events sunt utilizate pentru a captura actiuni care declanseaza o modificare in starea aplicatiei dezvoltate. Aceste evenimente sunt procesate si stocate in vederea analizarii schimbarilor produse in sistem. Cu alte cuvinte, domain events ajuta la exprimarea explicita a regulilor domeniului respectiv.

Implementand Domain Driven Design intr-un stil orientat pe obiecte, cand are loc o actiune asupra unui Aggregate, starea acelui aggregate se schimba, iar unul sau mai multe domain events sunt publicate. Problema care se pune in aceasta situatie este daca aggregate-ul ar trebui sa publice evenimentele in mod direct.

O posibila abordare este utilizarea unei clase statice care sa indice aparitia acesor evenimente imediat dupa ce aceasta clasa este apelata. In cazul utilizarii acestei metode, procesarea evenimentului de realizeaza imediat, facand in aceasta situatie testarea mai dificila, deoarece evenimentele care introduc efecte secundare sunt executate imediat, iar in momentul testarii focusul se afla pe ce se petrece in momentul respectiv in clasele curente, redirectionarea catre alte procesatoare de evenimente fiind problematica.

Aceasta abordare poate deveni prea complexa pentru o implementare corecta datorita acestor tranzactii neprevazute, motiv pentru care o metoda alternativa este utilizarea publicarii de evenimente in 2 pasi:

1. Crearea si inregistrarea evenimentelor in vederea publicarii
2. Transmiterea si procesarea evenimentelor colectionate dupa sau inaintea tranzactiei



Caragea Sebastian-Dacian,
Ingineria Sistemelor, anul 4, grupa 1

In figura de mai sus se poate observa un exemplu de sistem care are intrari printr-o interfata cu utilizatorul, un sistem de mesagerie, si o manipulare directa a unor tabele a bazelor de date. Pentru procesarea domain event-urilor exista componente in sistem care interactioneaza unele cu celelalte si convertesc intrarile intr-un sir de evenimente care sunt mai apoi stocate. Un procesator de evenimente le citeste mai apoi si declanseaza aplicatia care isi indeplineste rolul prestabilit. In acest sir de actiuni primul nivel al sistemului nu realizeaza nici o actiune in afara de crearea si stocarea de evenimente, in timp ce al doilea nivel poate ignora intrarile porpriu zise, el reactionand doar la eveniment.

In acest exemplu este prezentat un singur log de evenimente, desi in practica deseori aceste log-uri sunt separate daca evenimentele necesita diferite forme de raspuns. O interfata cu utilizatorul, spre exemplu, necesita o viteza de raspuns mult mai ridicata decat alte sisteme de mesagerie care nu se desfasoara in timp real, motiv pentru care cele doua situatii necesita log-uri si procesari diferite.

Deoarece evenimentele se petrec la un anumit moment in timp, de cele mai multe ori este necesar ca ele sa contina informatii legate de timp. Doi indicatori ai timpului sunt prezenti in stocarea evenimentului, timpul la care evenimentul a aparut, respectiv timpul la care evenimentul a fost observat.

Din punct de vedere semantic, domain events si integration events reprezinta acelasi lucru: notificari despre actiuni sau schimbari petrecute in sistem. Cu toate astea, domain events sunt doar mesaje transmise unui dispecer de evenimente. Pe de alta parte, scopul integration event-urilor este transmiterea si propagarea trazactiilor si actualizarilor catre subsisteme aditionale. Integeretion events trebuie sa fie bazate pe comunicare asincrona intre mai multe microservicii sau chiar subsisteme/aplicatii externe.

In concluzie, Domain events sunt utilizate pentru a explicita schimbarile dintr-un anumit domeniu care au loc intr-un sistem. Aceastea sunt utilizate pentru a obtine design, implementare si testare mai robuste. Diferitele parti ale unui sistem pot fi separate mult mai usor, fapt ce previne serviciile sau aggregate-urile de dimensiuni mari. De asemenea, acestea simplifica viitoarele integrari cu alte sisteme.