Tema 2 PSSC

Dezvoltarea de sisteme software extensibile folosind Managed Extensibility Framework

Având în vedere că pentru majoritatea companiilor, îndeplinirea cerinţelor funcţionale şi livrarea unui produs performant în timp util au în mod evident cea mai mare prioritate, o mare parte din sistemele software complexe exercită nevoia de a putea fi extinse cu noi funcţionalităţi, respectiv cu noi module. Pentru că proiectarea unui sistem software extensibil nu este uşor de realizat, au fost dezvoltate o serie de platforme care facilitează proiectarea acestor sisteme software. Una dintre cele mai noi astfel de platforme este Managed Extensibility Framework (**MEF**) care permite dezvoltarea de aplicații .NET extensibile.

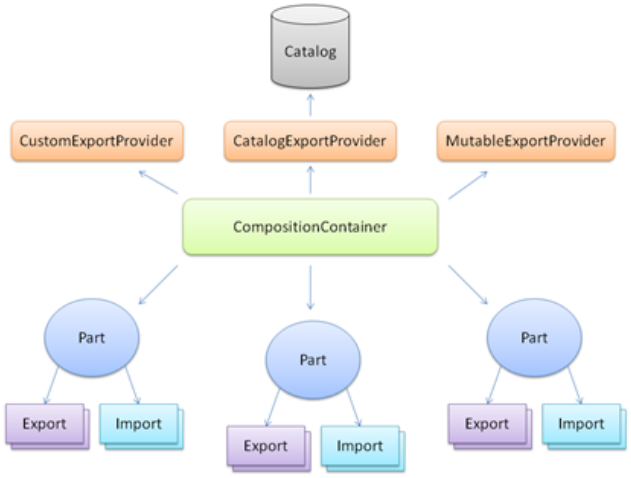

Fig. 1. Structura MEF

Prima componentă a structurii MEF este *catalogul*.

MEF oferă patru tipuri de cataloage, clasificare indusă de modalitatea prin care dorim să detectăm părţile:

1. ***DirectoryCatalog*** – reprezintă locul unde specificăm directorul din care dorim să incărcăm extensiile;
2. ***Type catalog*** – detectează toate părţile asociate unei colecţii de tipuri;
3. ***Assembly catalog*** – detectează toate părţile conţinute într-un ansamblu;
4. ***Aggregate catalog*** – ne permite să utilizăm o combinație de cataloage și să le grupăm, astfel încât să putem folosi atât un DirectoryCatalog cât și un AssemblyCatalog.

Miezul structurii MEF este *CompositionContainer*, care efectuează compunerea utilizând cataloage unde sunt încărcate toate părțile disponibile ce au fost detectate în sistem. Este responsabil pentru gestionarea compunerii prin crearea dependințelor dintre părți, asociind exporturile cu importurile.

Structura MEF se încheie cu așa numitele *parts*. Componentele constau în exportul și importul atributelor în fișierele lor, care ajută containerul să înțeleagă ce fișiere sunt consumatori (Importatori) și care fișiere sunt producători (Exportatori). Când se creează o parte, motorul compoziției MEF satisface importurile sale cu ceea ce este disponibil din alte părți.

Pentru a putea folosi platforma MEF este necesar să înțelegem anumite concepte:

- *Export:* - reprezintă un serviciu oferit de o parte;
  - aceeași parte poate exporta unul sau mai multe servicii;
- *Import:* - reprezintă un serviciu consumat de o parte;
  - o parte poate consuma unul sau mai multe servicii;
- *Contract:* - un contract este un identificator pentru un export sau un import;
- *Compunere:* – părțile sunt compuse de către MEF care le instanțiază și apoi face legătura între importuri și exporturi;

Problema extinderii

De exemplu, să ne imaginăm ca suntem arhitectul unei aplicații complexe care trebuie să ofere suport pentru extensibilitate. Aplicația noastră trebuie să includă un număr semnificativ de componente mai mici, fiind responsabilă de crearea și rularea acestora.

Cea mai simplă abordare a problemei este includerea componentelor ca și cod sursă în aplicația noastră și apelarea acestora direct din cod. Acest lucru are o serie de dezavantaje evidente, principalul fiind acela că nu putem adăuga componente noi fără a modifica codul sursă. Această restricție poate fi acceptată într-o aplicație Web dar nu poate fi utilizată într-o aplicație client.

O abordare puțin mai sofisticată ar fi furnizarea unei interfețe, pentru a permite decuplarea între aplicație și componentele sale. În acest mod, putem oferi o interfață pe care o componentă o poate implementa și un API pentru a permite interacțiunea cu aplicația noastră.

Deoarece aplicația nu dispune de capacitatea de descoperire a componentelor pe cont propriu, trebuie să se clarifice ce componente sunt disponibile și ar trebui încărcate. Acest lucru este de obicei realizat prin înregistrarea explicită a componentelor disponibile într-un fișier de configurare. Asigurarea faptului ca aceste componente sunt corecte devine însă o problemă de întreținere. În plus, componentele sunt incapabile să comunice una cu alta.

În locul acestei înregistrări explicite a componentelor disponibile, MEF oferă o modalitate de a le descoperi implicit, prin *compoziție*. Această abordare rezolvă problemele discutate în secțiunea anterioară.

Modelul MEF permite extensiilor să fie refolosite de la o aplicație la alta facilitând astfel dezvoltarea unui sistem de testare, independent de aplicație, pentru a testa componentele extensiei.

În concluzie, *parts* și *compositionContainer* reprezintă elementele de bază ale unei aplicații MEF. Fără MEF, orice aplicație care dorește să suporte un model de plugin trebuie să își creeze propria infrastructură de la zero. Aceste pluginuri vor fi adesea specifice aplicațiilor și nu pot fi refolosite în mai multe implementări.