Business engineering. Generic Software Architecture in an Object Oriented View

Lect.dr. Mihaela MUREŞAN
Universitatea Creştină „Dimitrie Cantemir", Bucureşti

The generic software architecture offers a solution for the the information system's development and implementation. A generic software/non-software model could be developed by integrating the enterprise blueprint concept (Zachman) and the object oriented paradigm (Coad's archetype concept). The standardization of the generic software architecture for various specific software components could be a direction of crucial importance, offering the guarantee of the quality of the model and increasing the efficiency of the design, development and implementation of the software. This approach is also useful for the implementation of the ERP systems designed to fit the user's particular requirements.
Keywords*: software engineering, software architecture, object oriented design, archetype, domain-neutral component*

Arhitectura unui sistem se materializează sub forma unui ansamblu de reprezentări care utilizează anumite convenţii, constituite ca meta-modele. Arhitecturile generice la nivel conceptual pentru diferite domenii de aplicaţie necesită instrumente cu ajutorul cărora să se poată construi modele generale, reutilizabile prin adaptare la un context dat. Utilizarea paradigmelor obiectuale permite reprezentarea viziunilor diferite ale actorilor implicaţi în procesele de dezvoltare a produselor software. Se remarcă de asemenea avantajele utilizării limbajului unificat de modelare (UML), limbaj specializat în vizualizarea, construirea şi documentarea sistemelor software, ca şi pentru modelarea proceselor de afaceri şi a sistemelor non-software. Pentru arhitecturile generice propuse, într-o abordare obiectuală se pot utiliza ca instrumente de reprezentare stereotipurile, şabloanele sau arhetipurile.

Pornind de la modelarea generică a proceselor de afaceri oferită de cadrul arhitectural al lui Zachman, şabloanele utilizate pot fi la nivelul proceselor de afaceri (şabloane ale proceselor de afaceri - arhitectura întreprinderii) sau la nivelul soluţiei informatice de gestionare a unei organizaţii, distingându-se şabloanele de arhitectură, de proiectare şi de implementare.

O soluţie chiar mai flexibilă şi cu un grad mai mare de generalitate este oferită de utilizarea arhetipurilor care permit realizarea unor arhitecturi software generice.

Concepte obiectuale implementate pentru dezvoltarea arhitecturilor generice software

O idee care poate fi utilizată cu succes în dezvoltarea arhitecturilor software este aceea a arhetipurilor. Arhetipurile reprezintă nuclee de structuri stabile care corespund mai mult sau mai puţin unui anumit domeniu, a căror proiecţie într-un context dat poate contribui la soluţionarea unei probleme în anumite condiţii particulare specificate. Datorită reprezentărilor sugestive cu caracter abstract, dar şi flexibil în acelaşi timp, arhetipurile par a fi instrumentele adecvate dezvoltării unor arhitecturi generice pentru procesele economice.

Arhetipurile reprezintă forme sau şabloane pentru un număr mic de categorii de clase. Acestea specifică atributele, legăturile, metodele, punctele de conectare şi de interacţiune care sunt tipice pentru clasele din acea categorie. Conceptul de arhetip a apărut ca o variantă cu o conotaţie mai amplă faţă de conceptul de stereotip din UML, care reprezintă un model invariabil pentru o categorie de clase. Arhetipul reprezintă o formă care este respectată *mai mult sau mai puţin* de toate clasele dintr-o anumită categorie [3]. Spre deosebire de caracterul invariabil al stereoti-

purilor, arhetipurile introduc un grad de sporit de flexibilitate, ceea ce permite dezvoltarea unor modele generice compuse din interconectarea arhetipurilor, componente care pot fi reutilizabile. Ideea centrală, avansată de Peter Coad şi colectivul său, este aceea de *componentă independentă de domeniu* (*domain-neutral component* [3]), reprezentând o matrice structurată pe mai multe dimensiuni. Prin generalizare, au fost identificate 4 tipuri de arhetipuri care interacţionează pentru a forma o *componentă independentă de domeniu*:

1.*arhetipul moment-interval*;
2.*arhetipul rol*;
3.*arhetipul intrare-catalog-descriere*;
4.*arhetipul terţ-loc-lucru*.

1. *Arhetipul moment-interval*. Acest arhetip este cel mai important deoarece orice proces de afaceri este determinat de timp. Arhetipul moment-interval reprezintă evenimente şi condiţii ce apar la anumite momente sau se desfăşoară într-un interval de timp.

2. *Arhetipul rol*. Arhetipul indică modul de participare al actorilor şi obiectelor în cadrul proceselor analizate. Se identifică astfel un nucleu generic referitor la atribute şi comportament, care surprinde caracteristicile participantului, indiferent de rolul pe care-l are la un moment dat. Se menţionează că un actor în cadrul proceselor poate fi un terţ (persoană fizică sau juridică), un loc sau un lucru şi, de asemenea, se precizează că fiecare participant poate avea roluri multiple. Cu ajutorul acestui arhetip se pot reda stările diferite ale unui obiect, terţ sau loc.

3. *Arhetipul descriere*. Rolul acestui arhetip este de a colecţiona valorile generice care caracterizează toate elementele corespunzătoare descrierii. Descrierile corespund unui terţ, loc sau lucru, reprezentând principalele caracteristici ale elementului.

4. *Arhetipul terţ-loc-lucru*. Acest arhetip indică actorii implicaţi în procesele de afaceri analizate, fiind în legătură directă cu arhetipul rol şi descriere. În cazul utilizării arhetipurilor se efectuează identificarea actorilor (arhetip terţ-loc-lucru), descrierea acestora (arhetip descriere) şi rolurile pe care le au (arhetip rol).

În acest mod, se realizează o stratificare a diferitelor aspecte ale datelor, separându-se mai multe nivele: identificarea actorilor, caracteristicile structurale şi comportamentale, rolurile posibile şi determinarea temporală a rolului.

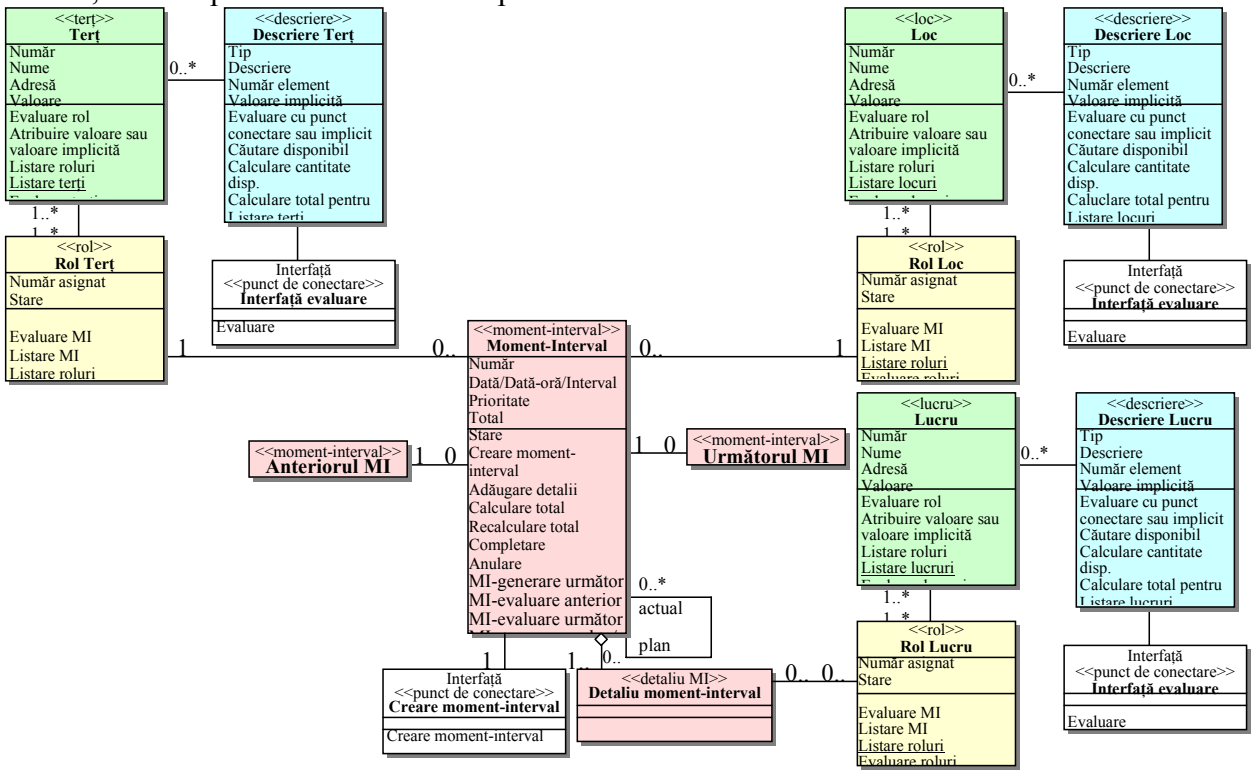

Fig.1. Componentă independentă de domeniu

Identificarea nivelelor este facilitată de utilizarea culorilor, astfel încât fiecărui arhetip îi este atribuită o anumită culoare, se evidenţiază mult mai uşor stratificarea şi totodată interacţiunea diferitelor tipuri arhetipale. Utilizarea culorilor subliniază simplitatea şi modul intuitiv de urmărire a diagramelor care alcătuiesc componenta independentă de domeniu. Culorile utilizate sunt următoarele:

- *roşu* - pentru arhetipul moment interval, ceea ce subliniază importanţa acestuia în determinarea comportamentului sistemului;
- *verde* - pentru arhetipul terţ-loc-lucru, evidenţiind actorii implicaţi;
- *albastru* - pentru arhetipul descriere, ceea ce sugerează stabilitatea acestui arhetip;
- *galben* - pentru arhetipul rol, ceea ce avertizează asupra multiplicităţii rolurilor unui actor în cadrul sistemului,
- *alb* - pentru elementele de interfaţă, evidenţiind modul de comunicare.

Modul de interacţiune a arhetipurilor este repetabil şi predictibil, ceea ce face posibilă realizarea unei componente generice (independente de domeniu), care ilustrează conectarea arhetipurilor şi legăturile dintre ele.

În cadrul componentei independente de domeniu se stabilesc legăturile posibile între toate tipurile de arhetipuri, care pot fi directe sau prin intermediul interfeţei. Legăturile directe presupun cunoaşterea structurii arhetipului cu care se realizează comunicarea. Legătura prin intermediul interfeţei nu necesită cunoaşterea structurii şi se stabileşte prin puncte de conectare. Posibile puncte de conectare sunt definite pentru arhetipurile descrieri (prezintă o operaţie generică de tip evaluare) şi momente-interval/ MI (prezintă o operaţie generică de tip creare MI). Utilizarea punctelor de conectare şi a interfeţelor sporeşte flexibilitatea modelului, spre deosebire de conectarea directă care conferă simplitate modelului. Structurarea componentei independente de domeniu (neutrale) este prezentată în figura 1.

Acest instrument independent de domeniu poate fi utilizat cu succes pentru realizarea unui model generic pentru procesele economice.

Structurarea componentelor independente de domeniu permite abordarea principalelor dimensiuni ale mediului de afaceri: timpul şi spaţiul. Sunt desemnate totodată categorii generice pentru principalii actori implicaţi şi pentru obiectele manipulate în sistem. Diagramele realizate astfel pot alcătui o arhitectură generică conceptuală pentru un anumit domeniu de aplicaţie, care apoi să poată fi transpusă într-un anumit context particular.

Pe lângă claritatea şi simplitatea reprezentărilor, arhetipurile prezintă o structurare adecvată cadrului arhitectural al lui Zachman pentru redarea proceselor de afaceri, ceea ce sporeşte accesibilitatea la modelele dezvoltate pe baza arhetipurilor. Corespondenţa între conceptele utilizate în cadrul de integrare a proceselor de afaceri introdus de Zachman şi cele folosite de arhetipuri este ilustrată în figura 2.

Prin structurarea pe mai multe dimensiuni, evidenţiindu-se aspectele temporale şi spaţiale, modelele dezvoltate pot constitui baza trecerii spre analiza proceselor (tehnologie OLAP – OnLine Analytical Processing), facilitând dezvoltarea modelelor pentru depozitele de date.

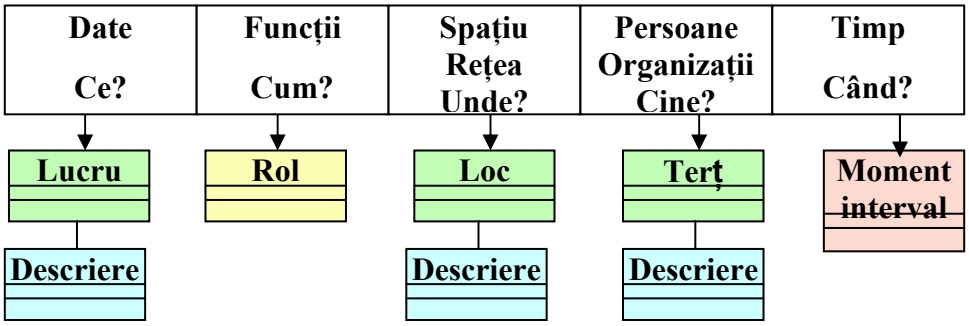

Fig.2. Utilizarea arhetipurilor pentru redarea arhitecturii Zachman

Redarea sintetică şi flexibilă a principalelor obiecte şi a interacţiunilor lor în determinarea lor temporală şi spaţială nu are intenţia de abordare exhaustivă a unui anumit domeniu, ci de a surprinde esenţa fenomenelor, care să constituie nucleul unor dezvoltări viitoare. Abstractizările oferite de arhetipuri asigură gradul de generalitate necesar arhitecturilor generice, facilitând totdată, prin funcţionalitatea lor, aplicabilitatea concretă.

O astfel de concepţie reprezintă o posibilitate de dezvoltare a unor sisteme unicat, adaptate perfect unui context dat, pe baza unor arhitecturi generice corespunzătoare principalelor obiecte şi comportamentului acestora într-un anumit domeniu de aplicaţie. Acest demers la nivel conceptual este gândit în termeni funcţionali, fiind utilizat la nivel operaţional. Utilizarea arhitecturii generice software se poate particulariza constituind baza dezvoltării unui sistem informatic integrat sau cadrul pentru adaptarea unui sistem de tip ERP (Enterprise Resource Planning) la cerinţele specifice ale utilizatorului. O astfel de abordare reduce considerabil timpul şi gradul de risc în implementarea sistemelor ERP, permiţând într-un mod eficient adaptarea la cerinţele particulare ale unei organizaţii.

Din acest punct de vedere, utilizarea arhitecturilor generice bazate pe arhetipuri, constituie coloana vertebrală a sistemului, care va fi remodelată pe baza cerinţelor specifice, ceea ce diminuează riscul neconformităţii cu necesităţile reale ale unei anumite organizaţii. Arhitecturile generice reprezintă construcţii utilizabile şi re-utilizabile care pot fi folosite în dezvoltarea evolutivă a softwareului. Există diferite strategii de utilizare a arhitecturilor generice, şi anume:

- utilizarea ca atare a componentelor generice;
- extinderea conţinutului arhitecturilor generice prin adăugarea de noi componente;
- extinderea capabilităţilor componentelor arhitecturii generice prin adăugarea de noi metode şi puncte de conectare, care sporesc flexibilitatea modelului.

Prin viziunea de ansamblu, ca şi prin legătura între modele, arhitecturile generice care utilizează arhetipuri devin instrumente ale managementului resurselor informaţionale şi tehnologice. Dezvoltarea şi utilizarea unui cadru arhitectural corespunde şi paradigmelor calităţii, deoarece implementează elemente generale, verificate care constituie o garanţie pentru calitatea produsului final. O altă facilitate oferită de această abordare, şi care răspunde, de asemenea, cerinţelor modelului calităţii, este evaluarea rapidă a impactului implementării unor schimbări. În acest fel, la nivelul acestor modele generice, se pot verifica şi valida schimbările solicitate, reducându-se substanţial riscul de eroare sau de neconformitate cu cerinţele.

Modelele generice bazate pe arhetipuri combină avantajele oferite de paradigmele obiectuale şi de limbajul de modelare unificat (UML) cu cele oferite de arhetipuri. Astfel, acestea, bazându-se pe concepţia orientată obiect, dispun, la nivelul intrării în sistem („front-office”), de o capacitate mai mare de a surprinde aspectele complexe ale realităţii, iar la nivelul intern al sistemului („back-office”) beneficiază de o interfaţă mai clară prin îmbinarea la nivel de obiect a datelor şi funcţiilor, asigurând o coeziune mai mare a componentelor şi o cuplare mai redusă între pachete.

Utilizarea în particular a arhetipurilor în dezvoltarea arhitecturii la nivel conceptual, facilitează descrierea realităţii prin structurarea şi generalizarea claselor îmbinate în componentele independente de domeniu. Arhetipurile implementează limbajul UML, ceea ce implicit le conferă posibilitatea de îmbinare a aspectelor formale, structurate cu cele descriptive. Se remarcă, totodată, simplitatea abordării care facilitează înţelegerea şi utilizarea modelului şi de către nespecialişti. O astfel de modelare sporeşte gradul de generalitate, extinzând posibilitatea de reutilizare a modelelor pentru dezvoltarea, prin completare şi particularizare, a mai multor produse software. Utilizarea arhetipurilor, prin marcarea distinctă a punctelor de conectare facilitează integrarea componentelor generice, dezvoltate sub formă de pachete.

O astfel de modelare reprezintă o provocare în surprinderea complexității lumii reale. Complexitatea creează o presiune asupra actorilor implicați în dezvoltarea software-ului prin numărul mare de detalii pe care acesta trebuie să le perceapă şi să le utilizeze la un moment dat, ceea ce depăşeşte abilitățile normale ale individului. Instrumentele de abstractizare oferite de arhetipuri permit tocmai structurări şi rafinări care facilitează surprinderea aspectelor complexe.
Arhetipurile, ca tehnică de modelare, nu asigură numai o abstractizare a informațiilor şi o redare a realității, ci şi o modalitate de găsire a soluțiilor tehnice pentru problema dată pe baza modelului construit.

2. Articularea arhitecturii generice în cadrul proceselor ingineriei software

Dezvoltarea sistemelor informatice bazate pe arhitecturi software reprezintă o etapă de maturizare a rolului pe care acestea îl au în cadrul proceselor ciclului de viață a produselor software. Utilizarea arhitecturii software asigură pe de o parte modelarea adecvată a realității, iar pe de altă parte soluții referitoare la dezvoltarea software-ului şi facilități de interpretare a soluțiilor oferite de model în contextul lumii reale.
Ideea unui nucleu arhitectural care să fie standardizat nu impietează inovația şi dezvoltarea în sens evolutiv a sistemelor, prin introducerea unei uniformități în abordare, ci reprezintă o tehnică care facilitează dezvoltarea unui sistem nou sau introducerea unor îmbunătățiri asupra unui sistem existent, prin oferirea unor soluții predefinite şi verificate. O astfel de practică poate fi inclusă în instrumentarul ingineriei şi re-ingineriei software, fiind, de asemenea, un element care poate contribui la dezvoltarea calității sistemului informatic ca o proprietate intrinsecă a acestuia.
Abordarea arhitecturii la nivel conceptual reprezintă concentrarea asupra proceselor „front end", ceea ce permite o reprezentare funcțională a cerințelor pentru o clasă de produse. Prin modelele construite se realizează o abstractizare pe mai multe nivele care asigură tranziția de la o problemă reală din cadrul proceselor de afaceri la o soluție software. Arhitectura generică surprinde cu uşurință viziunile diferite ale celor implicați în realizarea produselor software şi permite totodată integrarea acestor perspective. Se oferă astfel o bază a comunicării între utilizatorii şi specialiştii implicați în proiectarea şi dezvoltarea produsului software. Totodată, se asigură premisele eliminării riscului de neconformitate cu cerințele utilizatorului, încă din fazele incipiente ale dezvoltării sistemului.
Arhitectura generică software reprezintă şi o soluție care poate fi implementată în rezolvarea problemelor particulare la nivelul unui produs din clasa pentru care a fost dezvoltată aceasta. În acest mod, se remarcă caracterul bivalent al arhitecturii generice ca interfață între procesele de afaceri şi aspectele tehnice referitoare la realizarea unui sistem informatic. Acest aspect particular al arhitecturilor software este redat în figura 3.

Arhitectura software conceptuală are implicații pe parcursul întregului ciclu de viață a produsului software, şi anume, se dezvoltă în faza de concepere a sistemului, fiind utilizată în faza de construire a sistemului şi de tranziție a acestuia către o nouă versiune. Utilizarea arhitecturilor generice reprezintă, de asemenea, un suport de integrare între fazele proceselor ingineriei software.
Rolul arhitecturii generice în calitate de abstractizare a realității nu este numai descriptiv, acest instrument putând fi utilizat în mod operațional pentru verificarea, la nivel conceptual, a integrității, consistenței şi completitudinii design-ului. În acest mod, utilizarea arhitecturii generice facilitează identificarea neconcordanțelor modelului dezvoltat față de problema reală şi permite ajustarea rapidă a soluției conform necesităților.
Este important de subliniat caracterul reutilizabil al arhitecturilor generice în diferite contexte. Reutilizarea este posibilă datorită caracteristicilor acestora, dintre care, subliniem, în primul rând: stabilitatea şi adaptabilitatea.
Dezvoltarea modularizată, cu o cuplare destul de slabă a componentelor generice şi de-

limitarea clară a punctelor de cuplare sunt elementele care conferă stabilitate construcţiilor prezentate, inserarea unor modificări afectând un număr redus de elemente (clase şi interacţiuni). Această concepţie asigură totodată şi flexibilitate componentelor arhitecturale, permiţând o adaptare rapidă la noi cerinţe, generate fie de restructurarea proceselor de afaceri, fie de schimbări tehnologice sau de necesitatea îmbunătăţirii soluţiilor existente.

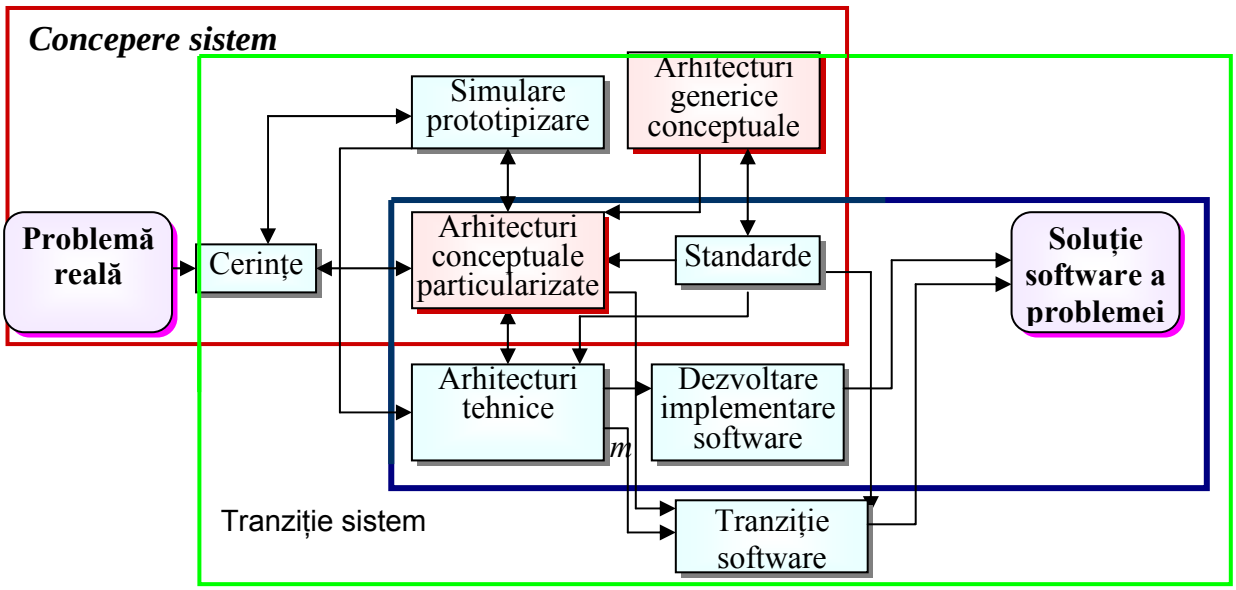

Fig.3. Poziţia arhitecturilor generice în cadrul proceselor ingineriei software

Creşterea complexităţii în domeniul proceselor de afaceri şi în domeniul tehnologiei informaţiei, ca şi de dinamica accentuată a schimbărilor în toate domeniile, accentuează necesitatea găsirii unor soluţii care să faciliteze înţelegerea şi redarea realităţii, să scurteze perioada de dezvoltare a produselor software şi să asigure stabilitate pe parcursul evoluţiei acestora. Arhitecturile generice, ca şi paradigmele obiectuale, reprezintă soluţii ale ingineriei software pentru depăşirea dificultăţilor specifice contextului actual.
Dezvoltarea arhitecturii generice oferă cadrul de înţelegere a unor sisteme mari şi complexe, prin dezvoltarea rapidă de modele abstracte care fac posibilă realizarea unui sistem informatic de calitate, precum şi modificarea ulterioară a acestuia. Este posibilă totodată şi reutilizarea acestor structuri generice pentru dezvoltarea altor produse sau pentru îmbunătăţirea produsului software dezvoltat. Datorită caracterului lor funcţional, structurile generice dezvoltate la nivel conceptual facilitează testarea rapidă a modificărilor, ceea ce diminuează riscul de inducere a unor erori la schimbarea sistemului.
Printr-o analogie cu domeniul programării, utilizarea arhitecturilor generice în analiza şi proiectarea proceselor/sistemelor are un efect similar cu saltul care s-a produs în programare prin introducerea generatoarelor cu componentele Wizard, care ofereau soluţii prefabricate, care corespundeau într-un anumit grad necesităţilor utilizatorului, putând fi uşor modificate, astfel încât să satisfacă pe deplin cerinţele acestuia. Arhitecturile generice software pentru diferite domenii de aplicaţie reprezintă tocmai soluţii prefabricate care pot fi incluse în bibliotecile de modele ale instrumentelor CASE (Computer Aided Software Engineering).

Succesul unor astfel de arhitecturi generice constă în coerenţa acestora, astfel încât ele constituie un nucleu universal pentru un anumit domeniu. Arhitecturile software se constituie într-o coloană vertebrală pentru construcţia unui sistem informatic, contribuind la maturizarea proceselor ingineriei sof-

tware. Dezvoltarea sistemelor informatice bazate pe utilizarea arhitecturilor software contribuie la restructurarea proceselor ciclului de viaţă a produselor informatice, şi în special a celor legate direct de utilizator (front-end). Astfel, se pot dezvolta abordări arhitecturale pentru un domeniu sau pentru o linie de produse, care pot fi apoi particularizate pentru un produs individual. Fiind construite ca modele abstracte, este posibilă reutilizarea facilă a structurilor arhitecturale, care pot fi uşor transferate de la un produs la altul.

Utilizarea arhitecturilor are implicaţii majore asupra calităţii produselor software. Articularea arhitecturilor software cu un sistem de atribute de calitate, permite evaluarea modelelor generice dezvoltate şi a modului de implementare a acestora, încă din fazele timpurii ale realizării sistemului. O mutaţie esenţială în abordarea arhitecturilor software este introdusă prin rafinarea tehnicilor de analiză a arhitecturilor în funcţie de varietatea atributelor de calitate şi a interacţiunii dintre acestea. În acest mod, arhitecturile generice contribuie la rezolvarea problemelor legate de calitatea sistemelor pe parcursul dezvoltării acestora.

Utilizarea modelelor generice în procesele ingineriei software prezintă o serie de avantaje, după cum urmează:

- promovează *integrarea datelor şi a aplicaţiilor*;
- facilitează *partajarea datelor şi a structurilor* de date, introducând descrieri abstracte (*TAD - Tipuri Abstracte de Date*) şi contribuind la reducerea redundanţei datelor;
- asigură un cadru funcţional pentru *verificarea şi testarea conceptuală a modificărilor*;
- facilitează *comunicarea la nivel intern şi extern*;
- oferă un suport pentru *integrarea sistemelor operative* (tehnologie *OLTP*), cu cele de *analiză/manageriale* (tehnologie *OLAP*);
- asigură *reducerea riscurilor şi a costurilor de dezvoltare şi mentenanţă* a sistemului;
- îmbunătăţesc *eficienţa sistemului* prin reducerea resurselor consumate;
- asigură *creşterea productivităţii* în dezvoltarea şi mentenanţa sistemului;
- contribuie la *dezvoltarea de standarde şi proceduri de lucru operaţionale*;
- contribuie la *calitatea sistemului*;
- facilitează *managementul sistemului şi al calităţii acestuia*.

Considerăm că dezvoltarea şi standardizarea arhitecturilor generice pentru diferite domenii reprezintă o provocare actuală nu numai pentru domeniul ingineriei software, ci şi pentru managementul întreprinderilor care se confruntă cu problemele legate de dominarea complexităţii şi dinamicii contextului în care acţionează. Standardizarea arhitecturilor generice are influenţe deci şi asupra ingineriei şi re-ingineriei afacerilor, depăşind cadrul software-ului.

Bibliografie

[1] Boehm Barry, (1995), http://www.sei.cmu.edu/architecture/definitions.html, 2002

[2] Booch, Rumbaugh, Jacobson - The UML Modeling Language User Guide, Addison-Wesley, 1999

[3] Coad Peter, Lefebvre Eric, De Luca Jeff - *Java modeling in color with UML*, Prentice Hall, 1999

[4] Cornion J., P., Haltab, N., J., *Qui a encore peur de l'informatique*, Eyrolles, Paris, 1990

[5] Fowler M., *Analysis Patterns – Reusable Object Models*, Addison Wesley, 2000, U.S.A.

[6] Hay David C., The Zachman framework:. An introduction

[7] Mehdi Jazayeri, Alexander Ran, Frank van der Linden - Software Architecture for Product Families: Principles and Practice,. Addison Wesley Longman, 2000

[8] Mureşan Mihaela, Arhitecturi şi standarde în dezvoltarea produselor software de contabilitate, Ed. Curtea Veche, Bucureşti, 2003

[9] Alan Perkins, *Implementing the Zachman Framework for Enterprise Architecture* - http://www.visible.com

[10] Zaharie Dorin, Roşca Ion, Proiectarea obiectuală a sistemelor informatice, DualTech, Bucureşti, 2002

[11] ***, First International Workshop on Architectures for Software Systems, Mary Shaw, 1995, http://www.sei.cmu.edu/architecture/definitions.html, martie 2002

[12] ***, IEEE 1471/ 2000, Recommended Practice for Architectural Description of Software-Intensive Systems