SOLUŢII DE INTEGRARE SOFTWARE A ACTIVITĂŢILOR ÎN INGINERIA CONCURENTĂ

prof. dr. ing. Mihai Aurelian Stanescu
dr. ing. Daniela Saru
dr. ing. Simona Liliana Caramihai

Universitatea POLITEHNICA din Bucureşti

Rezumat: Articolul prezintă paradigma ingineriei concurente şi, în special, un concept de bază al acesteia – echipa pluridisciplinară de proiectare. O astfel de echipă poate lucra eficient numai în contextul existenţei unei infrastructuri informatice adecvate, cu o arhitectură specifică. În cadrul articolului, sunt descrise principalele niveluri ale unei arhitecturi generice, evidenţiindu-se faptul că, două dintre acestea, necesită o proiectare " personalizată" în raport cu specificul fiecarei întreprinderi, în timp ce un al treilea nivel, cel de la bază, trebuie să fie o platformă software de integrare a aplicaţiilor *end-user*. Deoarece astfel de platforme sunt deja disponibile pe piaţă, în ultima parte a articolului este inclusă şi o scurtă descriere a uneia dintre cele mai performante implementări.

Cuvinte cheie: inginerie secvenţială, ingineria concurentă, ciclu de viaţă al produsului, echipe de lucru multidisciplinare, medii flexibile, sistem strategic C4 (CAD, CAM, CAE, CIM), integrarea aplicaţiilor, CAS.CADE.

1. Ingineria concurentă ca paradigmă de fabricaţie

Într-o economie orientată spre globalizare, a fi competitiv înseamnă a răspunde cerinţelor consumatorilor din ce în ce mai sofisticaţi şi mai exigenţi, pe pieţe de desfacere din ce în ce mai fragmentate şi mai imprevizibile. Aceasta impune transformarea şi personalizarea producţiei de masă, reducerea termenelor de fabricaţie şi a ciclurilor de viaţă a produselor, producţie la cerere (previzionară), aprovizionări «just in time», preţuri, calitate şi servicii competitive.

Evoluţia sistemelor de producţie este jalonată de definirea unor modele şi soluţii de referinţă, care încapsulează experienţa şi nivelul tehnologic specifice perioadei respective. Conţinutul şi convergenţa ideatică a acestor avansuri manageriale, tehnologice şi organizatorice sunt sintetizate în paradigme, destinate să clarifice, să orienteze şi să compatibilizeze eforturile de perfecţionare a sistemelor de producţie într-o perioadă dată. O astfel de paradigmă este definită ca fiind rezultatul interacţiunii între inovaţia tehnologică, tehnologia informaţiei, cerinţele pieţei şi evoluţia contextului socio-economic.

Astfel, concretizând prezentarea la ultima jumătate a secolului–în mod cert cea mai dinamică din punctul de vedere al evoluţiei –paradigmele de fabricaţie care, s-au succedat au fost: linia de transfer (anii '60), insulele de fabricaţie (prima parte a deceniului opt), modelul de automatizare flexibilă (sfârşitul acestui deceniu) - orientat pe reducerea timpului de reacţie la cerinţele pieţei prin flexibilizarea soluţiilor de automatizare. Automatizarea flexibilă este prima paradigmă care pune pregnant în evidenţă rolul informaticii în perfecţionarea sistemelor de producţie. Pe aceeaşi linie, la începutul deceniului 9, a apărut modelul fabricaţiei integrate cu calculatorul, care urmăreşte să valorifice progresele rapide în domeniul tehnologiilor informatice pentru integrarea activităţilor de bază ale întreprinderii: proiectare produse, proiectare procese, producţie. Semnificative pentru evoluţia concepţiei privind sistemele de fabricaţie sunt şi alte două paradigme de modelare, apărute în decursul deceniului 9: lean production–orientat pe adaptarea rapidă la cerinţele pieţei, pe baza utilizării conceptului «just in time» şi human-centered, orientat pe activitatea de grup *(team-work)* căruia i se deleagă întreaga responsabilitate, dar şi libertatea de acţiune, necesară pentru realizarea sarcinilor primite [4].

Situându-se în acelaşi context evolutiv, a apărut şi s-a impus în ultimii ani paradigma ingineriei concurente, care urmareşte să ridice la un nivel calitativ mai înalt toate îmbunătăţirile privind conceptul de fabricaţie, pe care le-au adus precursoarele sale şi, mai ales, să imbine în cel mai fericit mod două concepte iniţial opuse: fabricaţia integrată cu calculatorul şi, respectiv, fabricaţia human-centered [5].

Conform definiţiei date de [10] ingineria concurentă este o abordare sistematică a proiectării integrate şi concurente (simultane) a produselor şi a proceselor aferente, inclusiv a celor de fabricaţie şi a celor auxiliare. Abordarea este gândită pentru a permite producătorilor şi proiectanţilor să ia în considerare, încă din fazele preliminare ale proiectării, toate elementele ciclului de viaţă ale unui produs, de la concepţie până la livrare, incluzând aspecte legate de calitate, costuri, cerinţe ale utilizatorilor.

2. Inginerie secvenţială *vs.* ingineria concurentă

Ingineria secvenţială *(Sequential Engineering - SE)* reprezintă ingineria "clasică" în sensul secvenţierii fazelor ciclului de viaţă al produselor.

Mai precis, înainte de începerea oricarei faze, trebuiau cunoscute toate informaţiile legate de faza precedentă.

Înainte ca toate problemele legate de fabricarea unui produs să fie complet rezolvate, era necesară realizarea unor "bucle de interacţiune" între departamente, bucle prin care se încerca rezolvarea problemelor de schimb de informaţii între specialişti din diferite domenii, implicaţi în realizarea produsului. Durata ciclului de viaţă se estima conform formulei [1]:

$$T_{SE} = \S T_{cerinte} + T_{definire\ produs} + T_{definire\ procese} + T_{fabricatie\ si\ asamblare} + T_{livrare\ si\ asistenta\ tehnica} \ddagger R_{factor}$$

unde R_{factor} reprezintă factorul echivalent de repetare a cărui valoare pentru ingineria secvenţială este >> 2

Această abordare informaţională a făcut ca, în

- proiecte neexecutabile în condiţiile tehnologice ale întreprinderii
- toleranţe contradictorii ale subansamblelor aceluiaşi produs
- depaşirea termenelor estimate

Ingineria concurentă (Concurrent Engineering - CE) urmăreşte, în principal, scurtarea ciclului de viaţă al produselor prin suprapunerea parţială sau totală a anumitor activităţi. Conform celor reprezentate in figura1, durata totală a ciclului de viaţă pentru ingineria concurentă este:

$$T_{CE} = \S T_{cerinte} + T_{definire\ produs} + T_{definire\ procese} + T_{fabricatie\ si\ asamblare} + T_{livrare\ si\ asistenta\ tehnica} \ddagger R_{factor}$$

unde, de această dată, R_{factor} are o valoare < 1 (de regulă aprox.0.5).

Pentru ca o întreprindere să înceapă să implementeze (din punct de vedere managerial)

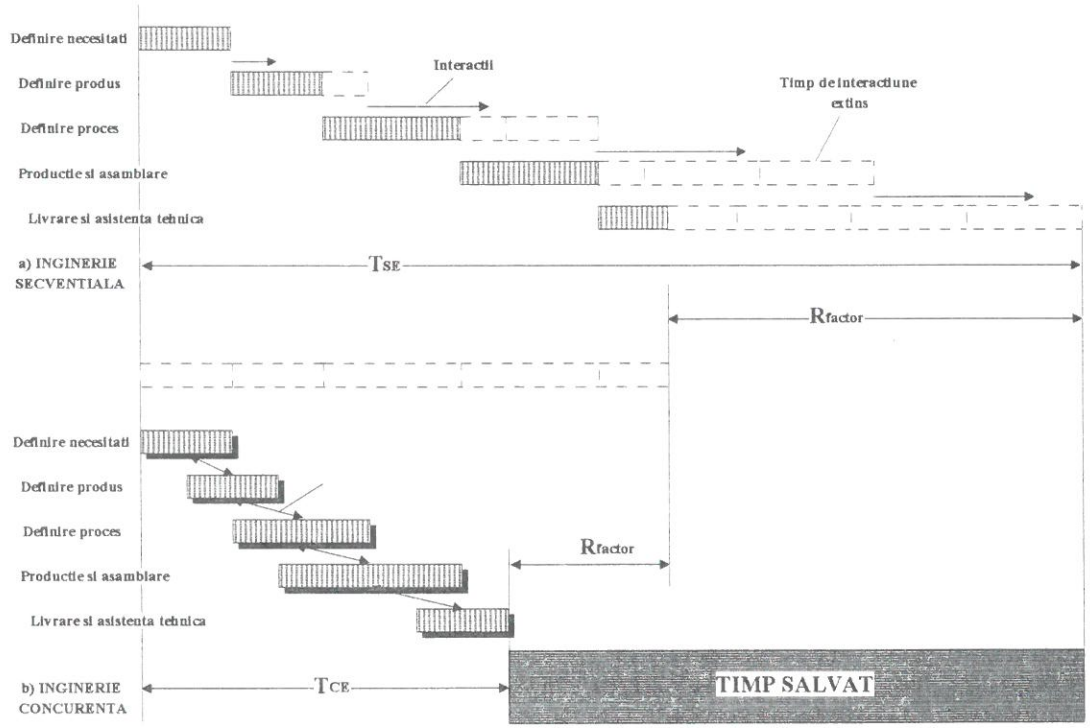

Figura 1

locul unei cooperări în vederea îndeplinirii scopurilor globale ale întreprinderii, să apară competiţii desfaşurate între diferitele departamente în vederea recunoaşterii importanţei fiecaruia în raport cu celelalte. Chiar în cazul obţinerii unor soluţii locale optime (în cadrul departamentelor), de cele mai multe ori aceste competiţii generau soluţii suboptimale la nivel global.

Principala problemă a ingineriei secvenţiale o reprezintă **lipsa comunicaţiei** on-line între diferitele departamente ale întreprinderii. Lipsa de comunicare provoacă disfuncţionalităţile cele mai frecvent remarcate în cazul fabricaţiei bazate pe inginerie secvenţială:

principiile ingineriei concurente se impune, însă, indeplinirea unei serii de condiţii preliminare şi anume [7]:

- stabilirea unor politici manageriale corespunzătoare (conducerea şi evaluarea proiectelor este total diferită în cadrul ingineriei concurente faţă de alternativa secvenţială) – cu asumarea completă a responsabilitatilor manageriale
- formarea de echipe de lucru multidisciplinare
- existenţa tehnologiilor şi a unei infrastructuri informatice integratoare corespunzătoare

- organizarea întreprinderii trebuie să fie flexibilă şi cu reactivitate mare

Prin contrast cu metodele ingineriei secvenţiale, unul dintre principiile de bază ale ingineriei concurente îl reprezintă **lucrul în echipă**. Structura de bază o reprezintă **echipa multidisciplinară,** dedicată realizării unui anumit produs. Figura 2 reprezintă o astfel de echipă multidisciplinară, incluzând specialişti de diferite formaţii şi lucrând într-un mediu care stimulează gândirea inovatoare. Atât furnizorii cât şi clienţii trebuie incluşi în echipă, contribuind la activităţi care ţin de realizarea produsului şi la alegerea maşinilor şi a instrumentelor necesare.

3. Soluţie generică pentru arhitectura de sistem. Multiplatforma din ingineria concurentă

Lucrul pe baza echipelor multidisciplinare presupune existenţa unei infrastructuri informatice aptă să asigure [6][8]:

- desfăşurarea de activităţi concurente de proiectare de către persoane de specialităţi diferite şi plasate în locuri diferite;
- actualizarea informaţiilor privind proiectul în timp real, pentru toţi membrii echipei;
- integrarea de aplicaţii şi formate de date diferite;
- dezvoltarea aplicaţiilor intr-un mediu flexibil;

Figura 3 prezintă o schemă bloc logică a unei arhitecturi informatice care răspunde acestor cerinţe.

Nivelul cel mai de jos este constituit din platforma de calcul, care reprezintă primul nivel al unei subarhitecturi de Inginerie Concurentă, fiind formată dintr-o platformă hardware, sisteme de operare şi aplicaţii. Aplicaţiile sunt instrumente care permit accesul transparent la baza de date globală şi la alte resurse ale sistemului. Au fost identificate trei tipuri de mecanisme [3][9] care stau la baza aplicaţiilor ce formează platforma de calcul:

1. Mecanismul Obiect - utilizat pentru gestionarea obiectelor, permite structurarea ierarhiilor complexe de obiecte aranjate în clase, subclase şi instante;
2. Mecanismul de Reguli -oferă membrilor grupului de lucru posibilitatea de a alege între diverse tehnici de proiectare specifice domeniilor de expertiza respective;
3. Mecanismul de Acces la Date folosit pentru maparea datelor dintr-o bază de date

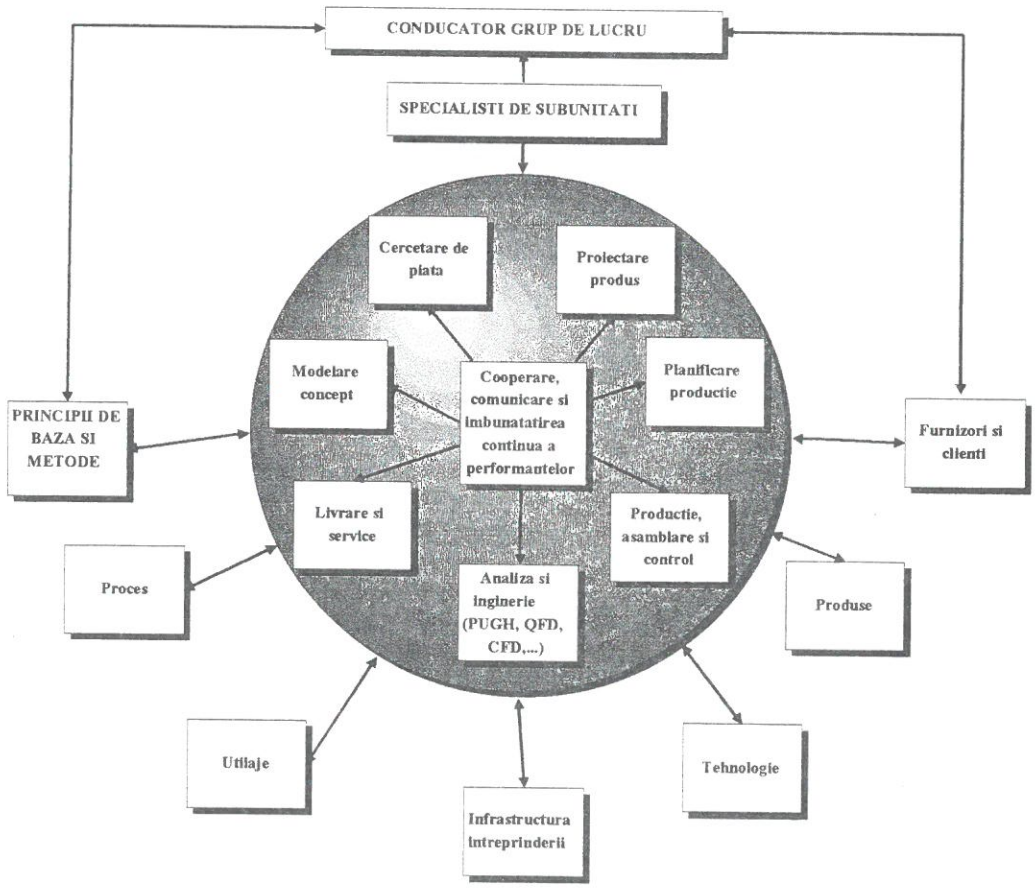

Figura 2.

relationalã în obiecte.

Al doilea nivel –interfaţa inteligentă oferă celor care dezvoltă aplicaţii o interfaţă program primară. Acest nivel constă dintr-un sistem C4 (CAD/CAM/CAE/CIM) strategic, instrumente generale şi instrumente de dezvoltare a aplicaţiilor (Figura 3). Nivelul de interfaţă inteligentă conţine

Interfaţa inteligentă trebuie să includă facilităţi de import/ export date, standarde de programare şi procesoare pentru limbaje de nivel înalt. Interfaţa inteligentă împreună cu limbajul de nivel înalt oferă o interfaţă de programare a aplicaţiilor (API) globală pentru echipa standard de lucru din ingineria concurentă.

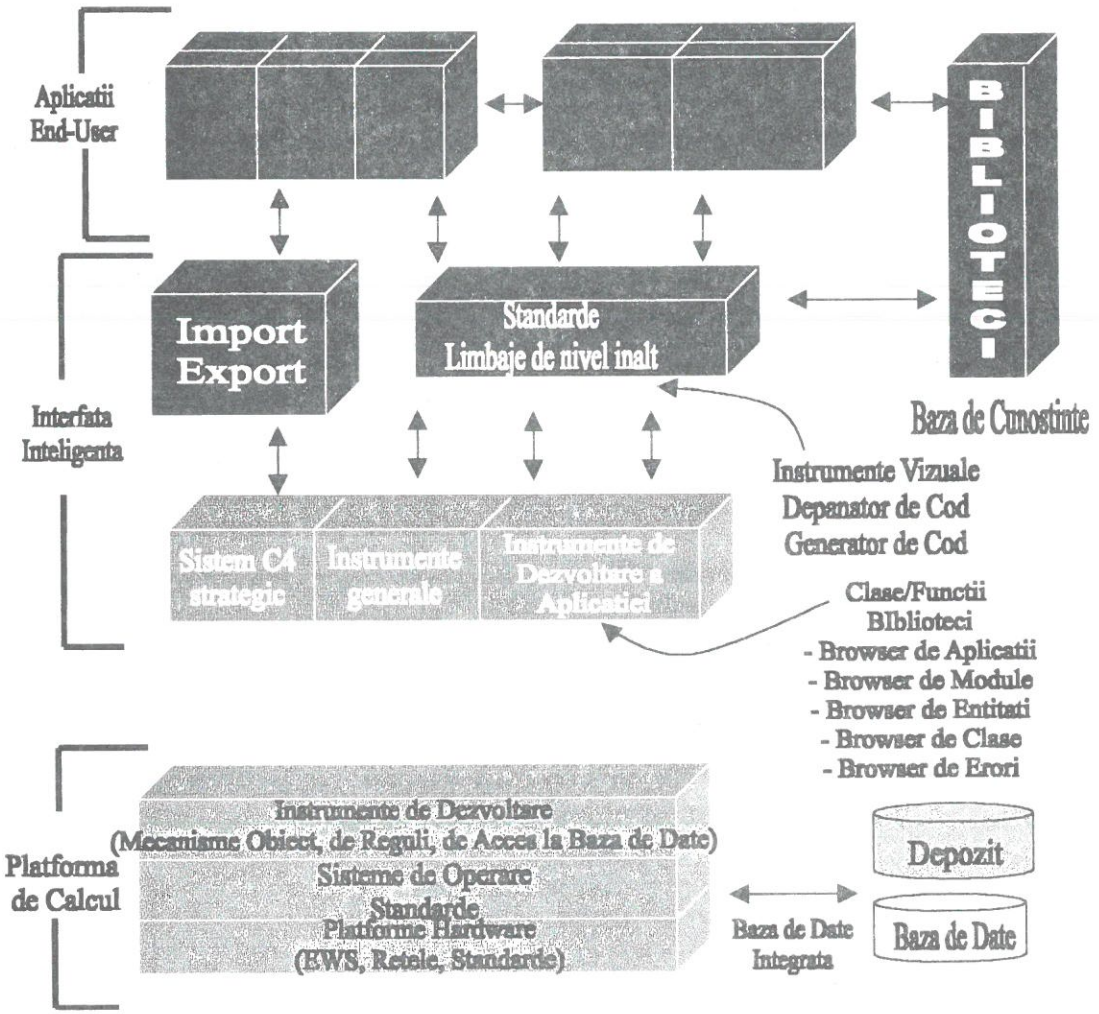

Figura 3.

trei componente principale:

1. Sistemul C4 strategic - conţine un set de utilitare C4 şi un set de facilităţi de construcţie a interfeţelor grafice end-user, având ca menire principală construirea unui model comun pentru baza de cunoştinte; acest sistem oferă o perspectivă a proiectării care este comună tuturor disciplinelor din cadrul Ingineriei Concurente;
2. Instrumente generale - mediul interactiv C sau C++ permite dezvoltarea de coduri sursa orientate obiect;
3. Instrumentele de dezvoltare a aplicaţiei - constituie o interfată pentru dezvoltarea bazelor obiect, a seturilor de reguli şi maparea bazelor de date intr-un mediu grafic.

Nivelul al treilea cuprinde aplicaţii end-user care comunică intre ele (orizontal) şi cu interfaţa inteligenta (vertical). Spre deosebire de primele două niveluri a caror proiectare şi implementare depinde de filosofia de lucru şi de organizarea intreprinderii pentru care trebuie să fie concepute, nivelul al treilea poate fi constituit dintr-o platformă integratoare [2] accesibilă comercial [11]. Un exemplu de astfel de platformă este aplicaţia CAS.CADE, furnizată de Matra Datavision [11].

4. Platforma de dezvoltare a aplicaţiilor în ingineria concurentă

Aplicaţia CAS.CADE este o platformă de dezvoltare, proiectare şi producere de aplicaţii

tehnice complexe [11]. Aplicațiile tipice, construite cu CAS.CADE, se ocupă, în general, de modelare plana și tridimensională și de organizarea datelor. De exemplu, sistemele CAD de uz general sau specializate sunt mult mai ușor de construit cu ajutorul platformei CAS.CAD.

CAS.CADE și ajută utilizatorul să organizeze produsele cât mai eficient;
- un set de instrumente de programare;

Toate aceste instrumente (unelte) pot fi folosite cu ajutorul unui set de comenzi UNIX, care formează *Workshop Organisation Kit (WOK).*

Structura aplicatiei CAS.CADE			
Biblioteci de obiecte	**Suportul pentru interfata**	**Suportul pentru baze de date**	**Mediul de dezv. (Software factory)**
-Modelare geometrica -Vizualizare -Clase de baza -Test de tip "Harness"	-Limbajul CCL -Comenzi de dialog -On-line Help -Vizualizare plana -Vizualizare tridimensionala	-Clase de persistenta -Vizualizare si modificare de baze de date (navigare) -Aplicatia de proiectare	-WOK -Administrarea codului sursa -Limbajul CDL -Utilitarul Make -Editorul Emacs

Figura 4. Produse CAS.CADE

CAS.CADE rezultă din integrarea celor patru componente importante ale platformei:

Bibliotecile de obiecte sunt componente orientate pe obiect, construite în C++, care au ca principală funcție modelarea geometrică. Bibliotecile de obiecte au fost construite astfel încât să fie într-adevăr modulare și extensibile. Oferă structuri de date pentru modelare geometrică, parametrizarea unui model, prezentare, afișare și selecție grafică. Algoritmii complecși și interfețele de nivel înalt sunt implementate drept clase separate. Structurile de date implementate în cadrul acestor biblioteci sunt folosite numai în timpul execuției, de către algoritmii de prelucrare. Pentru stocarea modelelor geometrice se folosesc bazele de date.

- **Suportul pentru interfață:** resurse care contribuie la crearea arhitecturii aplicației construite, inclusiv a interfeței cu utilizatorul.
- **Suportul de baze de date:** oferă facilități de stocare, care includ și o aplicație de administrare a bazelor de date.
- **Mediul de dezvoltare (Software Factory):** permite organizarea proiectului indiferent de dimensiunea echipei de proiectare. Oferă acces la celelalte trei componente și seteaza corespunzator instrumentele de dezvoltare.

Mediul de dezvoltare (Software Factory(SF)) permite membrilor unei echipe să colaboreze pentru realizarea unui proiect, indiferent de locul în care aceștia lucrează. CAS.CADE SF permite folosirea cât mai eficientă a componentelor software disponibile, precum a bibliotecilor de obiecte și a serviciilor interactive incluzând:

- instrumente de administrare care controlează ansamblurile făcute de

Acesta organizează mediul pe două direcții:

- un " depozit" pentru administrarea componentelor produselor software;
- mai multe " ateliere" destinate dezvoltării produsului.

" Depozitul" conține elementele componente ale produsului sau " unitățile de distribuție" *(Delivery Units)* (CAS.CADE este formată din aproximativ douăzeci de astfel de unități), prin intermediul cărora poate să comunice cu atelierele. Atelierele sunt constituite din mese de lucru *(workbench)* unde se realizează proiectele. Un atelier poate fi creat pentru a asambla o aplicație, pentru a dezvolta o nouă versiune a unui produs dat, pentru a o transfera la o altă platformă de lucru, etc.

" Depozitul" este o zonă de referință, în care sunt plasate unitățile de distribuție, fiind accesibil atelierelor din aceeași " uzină" *(Factory)*. În acest loc sunt instalate noile versiuni ale produselor și de aici atelierele își iau resursele necesare unui proiect.

Un " atelier"este o zonă pentru lucru de grup în care o echipă redusă indeplineste un ciclu complet de producție software. Atelierul conține o ierarhie a " meselor de lucru" bazată pe o singură " rădăcină"(masă de lucru principală). Unitățile de distribuție necesare sunt importate din depozit în unitățile de dezvoltare ale mesei de lucru principale. Mesele de lucru subordonate utilizează zone în care programatorii au acces la unitățile de dezvoltare ale mesei principale. Pot fi create mese de lucru intermediare care să folosească concomitent unitățile de dezvoltare.

Masa de lucru este o zonă privată, pentru utilizator, unde un singur programator realizează un proiect. Un proiect poate implica în lucru mai multe unități de dezvoltare, unul pentru fiecare componentă software ce urmează să fie creată. O sesiune de lucru începe prin

deschiderea unei mese de lucru care oferă acces la unităţile şi ca uneltele de dezvoltare corespunzătoare (compilatoare, utilitare etc.).

" Unitatea de dezvoltare" este cea mai mică unitate ce poate fi creată prin editare, compilare, editare de legături, testare. Ea este dedicată unui anumit tip de componentă software. Componentele principale sunt biblioteca de clase (care constă într-un set de clase C++), interfaţă, schema bazei de date, motorul, scriptul interactiv şi front-end-ul.

O " unitate de distribuţie" este un articol construit din mai multe unităţi de dezvoltare. Ea este realizată într-un atelier (în general la masa de lucru principală) şi stocată în depozit. Înainte să fie copiată în depozit, unitatea de distribuţie este reactualizată.

Cele mai folosite instrumente de programare pentru CAS.CADE sunt produse standard comerciale: compilator C++ , editor de legături, depanator, analizoare de performanţă etc. Aceste produse nu fac parte din pachetul CAS.CADE care furnizează însă toate instrumentele specifice cum ar fi limbajul de definire CAS.CADE (CDL), utilitarele şi editorul GNU Emacs.

Limbajul de definire CAS.CADE (CDL) este limbajul de definire a datelor, folosit la descrierea componentelor software C++ ale aplicaţiei (scripturile interactive nu sunt specificate în CDL). Acesta nu este un limbaj de programare, fiind folosit pentru a defini caracteristicile componentelor şi nu pentru implementarea lor (scrisă în C++). CDL este un limbaj orientat pe obiecte, specializat pe clase. El include clase abstracte, metode virtuale, supradefinirea, clase generice şi prezintă caracteristici precum supraîncarcarea, specificarea excepţiilor şi vizibilitatea atributelor (public, protejat, privat). Clasele înrudite fac parte dintr-un pachet care defineşte domenii de vizibilitate a numelor. Fişierele header C++ sunt generate automat din definitia CDL.

5. Concluzii

Articolul prezintă în mod sintetic paradigma ingineriei concurente, evidenţiind avantajele şi principalele caracteristici ale celor mai moderne metode de proiectare cu echipe pluridisciplinare. Este descrisă o arhitectură generică a infrastructurii informatice necesare, interacţiunile şi specificul fiecarui nivel în raport cu mediul în care urmează să fie utilizat. În final, se prezintă succint o platformă software de integrare a aplicaţiilor *end-user* comercială, care permite ilustrarea rolului şi a funcţiilor celui de-al treilea nivel al arhitecturii generice

Bibliografie

1. **CHEN, B., C-H. MENQ**: Initial Attempts On the Characterization of Functional Requirements of Mechanical Products. PED, Vol.59, Concurrent Engineering, ASME, edited by D. Dutta et al., Proc. of the Winter Annual Meeting of ASME, Nov. 8-13, 1992, Anaheim, C.A., pp. 315-329.
2. **CURRAN, L.**: STEP Bridges the Way to Better Product Modelling, Machine Design Vol. 66, No. 5, March 7, 1994, pp. 137-142.
3. **EDER, W.E.**: Design Science-Meta Science to Engineering Design. Design Theory and Methodology, ASME,-Vol 27, Sept. 16-19,1990 Chicago, IL., pp. 327-335.
4. **HOUPT, T.J.**: The Importance of Rapid Prototyping to Manufacturing and Integrated Product Development. Concurrent Engineering Office, Manufactuirng Tehnology Directorate, Wright Research and development Center, Wright-Patterson Air Force Base, Ohio, August, 1990.
5. **LARSEN, N.E. L. ALTING**: Dynamic Planning Enriches Concurrent Process Production Planning. International Journal of Production Research, Vol. 30, No. 8 August, 1992, pp. 1861-1876.
6. **PRASAD,B.,R.S. MORENC, R.M. RANGAN:** Information Management for Concurrent Engineering: Research Issues. Concurrent Engineering: Research & Applications. An International Journal, Vol. 1, 1993, pp. 1-19.
7. **PRASAD, B.**: Towards A Functional Design of A Concurrent Information Modelling System. Computer Modelling and Simulation in Engineering -An International Journal, Vol.1, No. 1,1996.
8. **PUGH,S.**, Total Design -Integrated Methods for Successful Product Engineering. Reading. MA: Addison-Wesley Publ. Company,1990
9. **STĂNESCU, A.M., DUMITRACHE I., CURAJ A.**.: Multi-mode based System Design within Virtual Factory Synergetic Framework, The 4th Int. Conf. on Concurrent Enterprising,1997.
10. **ULLRICH K.T, S.D. EPPINGER**: Product Design and Development, McGraw-Hill New-York,1994.
11. **WINNER R.I., PENNEL, J.P., BERTREND H.E., SLUSARCZUK**: *The Role of Concurrent Engineering in Weapons Systems Acquisintion* –IDA Report R-338, 1998.
12. *** CAS.CADE Users Guide, Matra Datavision, France, 1998.