CONSIDERAȚII ASUPRA MODELELOR DE DEZVOLTARE PENTRU PROIECTELE SOFTWARE DE RADIO-NAVIGAȚIE

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea Politehnica Timişoara
în domeniul INGINERIE ŞI MANAGEMENT
de către

Ing. Andrei Huțanu

Conducător ştiinţific: prof.univ.dr.ing. Gabriela Proștean
Referenţi ştiinţifici: prof.univ.dr in ec. ing. Dănuţ Dumitrașcu
prof.univ.dr. ing. Valentian Emilia Bălaș
prof.univ.dr.ing. Anca Drăghici

Ziua susţinerii tezei: 21.12.2016

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor
12. Ingineria sistemelor
13. Inginerie energetică
14. Calculatoare şi tehnologia informaţiei
15. Ingineria materialelor
16. Inginerie şi Management
17. Arhitectură
18. Inginerie civilă și instalații

Universitatea Politehnica din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Cuvânt înainte

Teza de doctorat a fost elaborată pe parcursul activității mele în cadrul Departamentului de Management al Universității Politehnica Timișoara.

Experiența acumulată în ultimii 14 ani în diferitele companii din domeniul automotive, precum și analiza efectuată pe parcursul activității mele în cadrul Departamentului de Management al Universității Politehnica din Timișoara, au dus la realizarea acestei teze de doctorat.

Scopul cercetării îl constituie identificarea soluțiilor practice la problemele de organizare și desfășurare ale proiectelor din domeniul automotive. Complexitatea din ce în ce mai mare a proiectelor automotive, precum și așteptările sporite din partea clienților au dus la elaborarea unor modalități de adaptare a proceselor actuale, prin inserarea de metodologii flexibile la modificări.

Elaborarea modelelor noi de dezvoltare impune cunoașterea detaliată a managementului de proiect, particular domeniului de desfășurare a proiectului precum și a modelelor de dezvoltare utilizate. Acumularea datelor în urma analizei impactului modificării cerințelor, precum și a diferitelor modele de dezvoltare a generat concluzia elaborării unui nou model de dezvoltare, prin inserarea metodologiilor agile în structura modelelor tradiționale de dezvoltare. La rândul lor, inserarea metodologiilor agile în modelele de dezvoltare traditionale in „V” aduc cu sine avantajul flexibilității necesare în dezvoltarea produselor inovatoare.

Aplicarea modelului este recomandată mai ales datorită flexibilității modului de abordare asupra modificării specificațiilor inițiale. Validarea modelului de dezvoltare a dus la concluzia că pe lângă economia de timp generată, noul model(2JCS) a adus avantaje financiare companiilor automotive.

Timişoara, Octombrie 2016 Andrei Huţanu

Ca in orice proiect de management am avut ocazia să experimentez suișurile și coborâșurile tipice activității de cercetare. Doresc să mulțumesc în primul rând conducătorului de doctorat, Prof. Univ. Dr. Ing. Gabriela Proștean pentru răbdarea, timpul și sfaturile acordate pe parcursul cercetării științifice.

Totodată doresc să mulțumesc în mod deosebit domnului Prof. Univ. Dr. Ing. Dumitru Mnerie pentru sfaturile acordate pe întreaga durată a studiului doctoral. Viteza de reacție și răspuns a membrilor comisiei de îndrumare la cererile de recenzie, în persoana domnilor Prof. Univ. Dr. Ing. Dumitru Mnerie, Prof. Univ. Dr. Ing. Ioan Filip și Conf. Dr. Ing. George Belgiu merită mulțumiri speciale.

Nu în ultimul rând doresc să mulțumesc familiei mele pentru sfaturile, răbdarea și înțelegerea de care au dat dovadă mai ales în momentele dificile din perioada studiului doctoral.

Huţanu, Andrei

Consideraţii asupra modelelor de dezvoltare pentru proiectele software de radio-navigaţie

Teze de doctorat ale UPT, Seria 16, Nr. 25, Editura Politehnica, 2016, 166 pagini, 64 figuri, 20 tabele.

ISSN: 2343-7928

ISSN-L:2343-7929

ISBN: 978-606-35-0100-5

Cuvinte cheie: eşalonarea calendaristica, tehnici de management, modele tradiționale si AGILE de dezvoltare software, sisteme automotive.

Rezumat,

Schimbarea certă a specificațiilor în timpul dezvoltării proiectelor automotive, precum și impunerea finalizării proiectului în eșalonarea calendaristică inițială, constrânge deseori coordonatorii de proiect să decidă reducerea conținutului unui proiect fără a ține cont de efectele unei astfel de decizii.

Companiile din domeniul automotive se văd astfel nevoite să își focalizeze atenția pe prioritizarea funcțiilor, pe care trebuie să le îndeplinească un sistem automotive în detrimentul implementării noilor tehnologii. Acumularea datelor în urma analizei impactului modificării cerințelor precum și a diferitelor modele de dezvoltare a generat concluzia elaborării unui nou model de dezvoltare (2JCS) prin inserarea metodologiilor agile în structura modelelor tradiționale de dezvoltare.

Utilizarea noului model de dezvoltare (2JCS), conceput in cadrul tezei, va permite proiectului să se încadreze în eșalonarea calendaristică inițială. Prin încadrarea proiectului în durata inițială de timp și datorită scăderii cuantumului costurilor prin neinițierea unui proiect nou de portare a cerințelor de actualizare (cum se realizează în mod inerent prin aplicarea modelul în V), s-au asigurat beneficii atât temporale, financiare cât și avantaje legate de activitățile de marketing.

CUPRINS

Notații, abrevieri, acronime

Notații, abrevieri, acronime	**Explicație**	**Limba**
OEM	Original Equipment Manufacturer	EN
PMBOK	Project Management Body of Knowledge	EN
GSM	Global System for Global Communication	EN
UML	Unified Modeling Language	EN
ASD	Adaptive Software Design	EN
FDD	Feature Driven Development	EN
DSDM	Dynamic System Development Methodology	EN
2JCS	Denumire model de dezvoltare (**J**AD – **J**AD - **C**olaborare ASD-**C**rystal Clear – **S**CRUM-Testare **S**CRUM)	RO
SPICE	Software Process Improvement and Capability Determination	EN
JAD	Joint Application Development	EN
PMI	Project Management Institute	EN
PRINCE	PRojects IN Controlled Environment	EN
ICB	International Competence Baseline	EN
OGC	Office of Goverment Commerce	EN
TOC	Teoria constrângerilor	RO
HAZOP	Hazard si Operabilitate	RO
NPF	numărul de persoane care lucrează în această fază	RO
NOE2	numărul de ore economisite în urma aplicării modelului 2JCS	RO
VS	valoarea orei de lucru a specialiștilor acestei faze	RO
ASD	Adaptive Software Development	EN

XP	Extreme Programming Development Methodology	EN
FDD	Feature Driven Development Methodology	EN
SCRUM	Software Capability Rational Unified Model	EN

Lista de figuri

Lista de tabele

1. INTRODUCERE

1.1. Actualitatea temei de cercetare și motivația autorului în alegerea ei

Atingerea obiectivelor proiectelor de dezvoltare software depinde în foarte mare măsură de eficacitatea analizei proceselor software, utilizate în dezvoltare, precum și de promptitudinea, cu care aceste procese se adaptează la necesitățile domeniului în care sunt aplicate. Teza actuală are ca scop analiza proceselor software folosite în dezvoltarea proiectelor din industria automotive, identificarea slăbiciunilor acestora precum și găsirea și aplicarea soluțiilor de contracarare incapacităților proceselor consacrate.

Mobilitatea persoanelor, precum și deținerea și răspândirea informațiilor au devenit elementele esențiale în evoluția societății. În consecinţă, dezvoltarea sistemelor de radio-navigație a devenit o necesitate, fiind un mediu care reușește să îmbine latura mobilă cu cea a informațiilor. Ultimele evoluții ale sistemelor de radio-navigație au prezentat elementele integrative și de conectivitate cu sisteme externe automobilului, mai concret telefoanele inteligente sau componente IT.

Particular industriei automotive, proiectele software sunt în strânsă legătură cu procesele de producție. În consecință, pentru fiecare tip de proiect se utilizează doar anumite modele de dezvoltare și procese software, în defavoarea altora, selecția fiind realizată în funcție de dimensiunea proiectelor software și de influența proceselor de producție asupra domeniului.

Procesele clasice utilizate în dezvoltarea proiectelor software automotive nu acoperă toata gama actuală de cerințe, necesară desfășurării proiectelor, fapt ce impune o analiză atentă a acestora, precum și a cauzelor care duc la eșecul proiectelor. Mai concret, din experienţa autorului cerințele unui proiect de dezvoltare software în radio-navigație au avut o creștere a complexității în ultimii ani după cum urmează:

- La mijlocul deceniului trecut au apărut cerințe de integrare a funcționalităților senzorilor de parcare și a camerei video;
- Începând cu anul 2005 au apărut cerințe diverse de implementare a protocoalelor de comunicare cu componentele externe automobilului, ca de exemplu protocoale de comunicare cu telefoanele mobile.
- Primele cerințe legate de integrarea agregatelor electrice, precum și afișarea datelor acestora în sistemele de radio-navigație au apărut la sfârșitul deceniului trecut.
- În jurul anului 2010 au apărut cerințe concrete de integrare și conectivitate la sistemele IT cu scopul afișării informațiilor în timp real prin intermediul internetului.

Problema stringentă, cu care se confruntă aceste proiecte la ora actuală, este cea a încadrării proiectelor în eșalonarea calendaristică inițială, datorită schimbărilor frecvente, care apar pe parcursul dezvoltării acestora. Schimbările, care bulversează eşalonarea calendaristică se datorează diferenței dintre ciclul de

viață al componentelor periferice integrate și cel al proiectelor automotive de radio-navigație, primul fiind de trei ori mai mic. Proiectele de radio-navigație au un ciclu de dezvoltare mult mai mare decât componentele periferice, apariția noilor tehnologii face ca înaintea finalizării fazei de introducere a produsului pe piață, acesta să fie considerat depășit tehnologic. Goldratt [38] afirma că ciclul de viață a proiectelor nu mai corespunde unei curbe clasice, aceasta curbă transformându-se în profilul unei lame de fierăstrău.

În realitate, modelele de implementare existente au un spectru general, fiind necesară analiza detaliată a etapelor acestora pentru a putea fi adaptate în mod particular la fiecare proiect automotive.

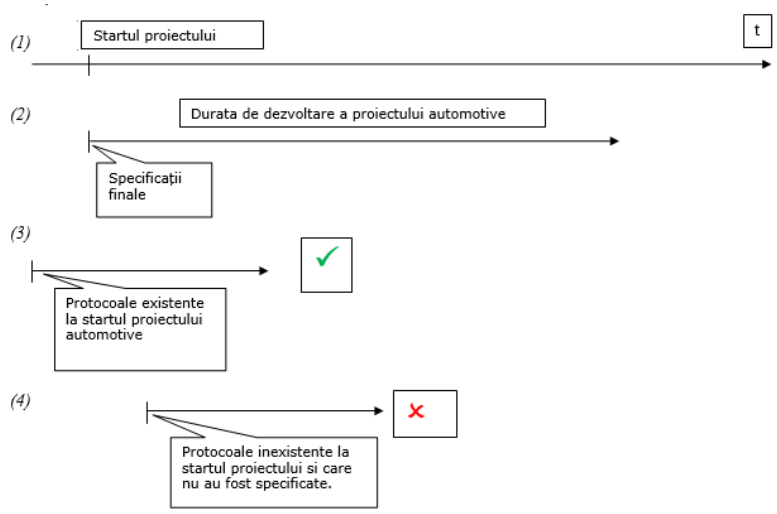

Fig. 1. 1 Evoluția dezvoltării protocoalelor de comunicare în proiecte automotive

Adaptarea proiectelor automotive la „mediul înconjurător" a dus la necesitatea adaptării cerințelor pe parcursul dezvoltării proiectelor. De altfel instabilitatea cerințelor în proiectele software este confirmată în literatura de specialitate. Necesitatea conectării componentelor periferice sau a sistemelor „after market" la sistemele de radio-navigație integrate ale automobilelor aduce instabilitate suplimentară asupra cerințelor proiectelor, generată în principal din necesitatea de adaptare a protocoalelor de comunicare. Specificarea tuturor cerințelor proiectului încă din faza inițială a acestuia este imposibilă din cauză că durata de dezvoltare a componentelor integrate este mult mai mare decât a componentelor periferice. De exemplu în proiectele automotive durata de dezvoltare a unui sistem de radio-navigație este de aproximativ patru ani, iar componentele periferice (ca de exemplu telefoanele inteligente sau componentele IT utilizate integrate în automobil) au o durată de dezvoltare mult mai scurtă, de aproximativ 6-12 luni.

Fig. 1.1 prezintă situația unui proiect automotive descris anterior, în care în starea inițială nu se cunosc toate datele necesare specificării tuturor cerințelor proiectului. Pe axa timpului (1) și în starea inițială a proiectului (2) se cunosc doar acele date tehnice care au devenit un standard pe piață (3) în acel moment. Cu cât proiectul avansează, apariția altor sisteme periferice constrânge dezvoltatorii de

sisteme automotive să găsească soluții pentru implementarea noilor protocoale de comunicare.

1.2. Scopul cercetării

Modificarea cerințelor în proiectele automotive precum și constrângerile de timp ale integrării acestora pe parcursul derulării proiectului au definit tema actuală de cercetare. Scopul principal al lucrării este de a identifica soluțiile practice pentru problemele de organizare și desfășurare ale proiectelor din domeniul automotive. Utilizarea modelelor tradiționale de dezvoltare nu permit implementarea ulterioară a specificațiilor. Prima măsură luată de coordonatorii de proiect pentru atingerea obiectivelor proiectului este reducerea conținutului acestuia. Chiar dacă în prima fază această măsură face ca obiectivele imediat următoare să poată fi atinse, aceasta generează costuri suplimentare companiilor prin necesitatea continuării sau creării de noi proiecte care să „completeze" vechile proiecte cu cerințele încă neimplementate. ***Aceste argumente au dus la definirea obiectivului principal al tezei, și anume, prezentarea modalităților de exploatare corectă a resursei de timp, obiectiv concretizat prin crearea unui nou model de dezvoltare, care să fie flexibil la schimbări și care să corespundă cererilor industriei automotive.***

Atingerea obiectivului general presupune parcurgerea următoarelor obiective intermediare:

- Analiza profundă a managementului de proiect pornind de la istoricul acestuia până la analiza critică a managementului de proiect pentru sistemele de radio-navigație;
- Analiza avantajelor și dezavantajelor utilizării ciclurilor de viață tradiționale și a metodologiilor AGILE în proiecte;
- Analiza impactului utilizării teoriei constrângerii dezvoltata de E.M. Goldratt aplicată în proiectele automotive;
- Analiza impactului utilizării teoriei hazardului (HAZOP) în proiectele automotive;
- Conceperea unui model de dezvoltare pentru proiectele software care să fie flexibil la schimbări și care să corespundă criteriilor industriei automotive;
- Validarea modelului conceput prin analiza comparativă a modelului tradițional cu noul model conceput în dezvoltarea proiectelor automotive.

1.3. Structurarea tezei de doctorat

Teza de doctorat este structurată în 6 capitole, în care autorul își propune să trateze in mod detaliat obiectivele intermediare, care împreuna, conduc la soluționarea obiectivului general al tezei. Fiecare capitol se finalizează cu soluții și concluzii specifice, capitolul final prezentând în sinteză concluziile lucrării, soluțiile și contribuțiile originale. În sinteză, lucrarea este organizată în felul următor:

Capitolul 1 prezintă actualitatea și motivarea alegerii temei de cercetare, obiectivele tezei de doctorat și structurarea acesteia.

Capitolul 2, structurat în două părți, sistematizează managementul de proiect, particularizând aspecte specifice managementului proiectelor automotive de radio-navigație.

Din analiza critică asupra managementului de proiect în general și a managementului de proiect a sistemelor de radio-navigație în particular au rezultat următoarele aspecte importante care stau la baza desăvârșirii tezei actuale de cercetare:

- atingerea obiectivelor proiectelor presupune cunoașterea detaliată a fazelor, pașilor proiectelor, și a ordinii în care aceștia trebuie să se desfășoare, precum și a factorilor critici specifici tipului de proiect, sinteză realizată în prima parte a capitolului;
- prin cercetarea literaturii de specialitate care abordează profund aspectele managementului de proiect a rezultat importanța factorilor critici stau la baza succesului proiectului. În domeniul automotive acest lucru este posibil, având în vedere faptul că literatura de specialitate abordează profund aceste aspecte;
- standardizarea proceselor utilizate în proiectele automotive, sub forma proceselor Spice, ajută coordonatorii de proiect în organizarea și luarea deciziilor pentru metodele de control și comandă a acestor tipuri de proiecte- aspect prezentat in ultima parte a capitolului;

Capitolul 3 prezintă caracteristicile generale ale ciclurilor de viață ale proiectelor de dezvoltare software. Capitolul debutează cu descrierea modelului tradițional de dezvoltare în „V", specific proiectelor de dezvoltare software. Modelul în „V" reprezintă o abordare superioară a altui model de dezvoltare tradițional numit „în cascadă". Cicluri de viață tradiționale, utilizate în managementul proiectelor software nu diferă în esență, acestea deosebindu-se doar în detaliu.

Inflexibilitatea modelelor tradiționale de dezvoltare în cadrul proiectelor de dezvoltare software au condus la dezvoltarea metodologiilor AGILE.

În cea dea doua parte a capitolului sunt analizate principalele metodologii AGILE, după cum urmează: Adaptive Software Development (ASD), Crystal, Feature Driven Development (FDD), Agile Model Driven Development (AMDD), SCRUM, Dynamic System Development (DSDM), Lean Development și Extreme Programming. Analiza critică realizată asupra tuturor modelelor de dezvoltare a scos în evidență faptul că metodologiile AGILE reușesc să elimine slăbiciuni ale metodelor tradiționale, prin accentul pus pe comunicarea și armonizarea echipelor ori de câte ori apare o perturbație în proiect, introducând-se astfel o reală flexibilitate în abordarea schimbărilor;

Capitolul se finalizează prin concretizarea obiectivului principal al acestuia, mai concret, acela de scoatere în evidentă în mod detaliat a aspectelor particulare ale modelelor de dezvoltare a proiectelor automotive cât și efectele asupra calendarului proiectului în cazul implementării modificării cerințelor în diferitele faze ale proiectului.

Capitolul 4, prin obiectivul său declarat, urmărește conceperea unui nou model de dezvoltare care să satisfacă cerințele proiectelor automotive de înaltă tehnologie. La baza implementării noului model de dezvoltare stau trei factori esențiali în succesul proiectelor sistemelor de radio-navigație. Primul factor este dat de decizia creării de noi tehnologii de vârf respectând calendarul proiectului. Al doilea factor îl reprezintă analiza continuă a modelelor tradiționale de dezvoltare în

cazul schimbării cerinţelor. Al treilea factor este reprezentat de optimizarea continuă a ciclului de viaţă a proiectelor software, aplicând metodele moderne ca de ex. „Teoria Constrângerilor" dezvoltată de E.M. Goldratt, care oferă explicaţii asupra neajunsurilor proceselor tradiţionale în dezvoltarea proiectelor software. Explicaţia asupra neajunsurilor proceselor tradiţionale în dezvoltarea software poate fi nuanţată prin sinteza factorilor de risc şi decizionali care duc la implementarea unei cerinţe noi sau modificate cu scopul propunerii unei soluţii astfel încât calendarul proiectului să nu fie influenţat;

În continuare se prezintă o metodă de calcul pentru validarea schimbării, care integrează toţi factorii decizionali. De asemenea, este prezentat modul de organizare a proiectelor în faza schimbării de cerinţe, fiind identificate efecte ale acelor schimbări în diferitele faze ale proiectelor bazate pe modelul in V.

În finalul capitolului autorul concepe şi elaborează un nou model de dezvoltare (2JCS) a proiectelor software de radio-navigaţie, având la bază rezultatele analizei diferitelor modele de dezvoltare clasice precum şi a metodologiilor agile.

Capitolul 5 realizează validarea modelului conceput pe două direcţii. Prima direcţie vizează încadrarea proiectelor de radio-navigaţie în eşalonarea calendaristică iniţială, iar cea de-a doua direcţie de validare vizează factori de măsurare a eficienţei economice.

Criteriile principale de validare sunt inspirate din criteriile principale de succes ale proiectelor, performanţa timpului şi cea financiară.

Validarea modelului 2JCS s-a realizat printr-o analiză comparativă între două firme: firma A în care a fost implementat modelul 2JCS şi firma V în care a fost utilizat modelul tradiţional în „V". Odată cu respectarea duratei de finalizare a proiectului în firma A, s-a obţinut o mai bună organizare a echipelor de lucru. Chiar dacă anumite faze de dezvoltare au depăşit bugetul declarat iniţial, s-a asigurat un beneficiu, datorită încadrării proiectului în durata iniţială de timp şi datorită scăderii cuantumului costurilor prin neiniţierea unui proiect nou de portare a cerinţelor de actualizare (cum se realizează în mod inerent prin aplicarea modelul în V).

Capitolul 6 prezintă concluziile şi contribuţiile personale ale autorului tezei. Sunt de asemenea enunţate direcţiile viitoare de cercetare ale temei, utilizând modelul de dezvoltare propus în această cercetare.

2. MANAGEMENTUL DE PROIECT, ASPECTE PARTICULARE DEZVOLTĂRII PRODUSELOR DE ÎNALTĂ TEHNOLOGIE.

Capitolul de față, structurat în două părți, sistematizează managementul de proiect, particularizând aspecte specifice managementului proiectelor automotive.

Prima parte prezintă în mod detaliat fazele managementului de proiect.

Cea de-a doua parte prezintă factorii critici ai proiectului precum și aspectele particulare dezvoltării sistemelor de radio-navigație, respectiv monitorizarea furnizorului, managementul lansării produselor, analiza cerințelor, integrarea sistemului și testarea acestuia, managementul de proiect, managementul riscului, toate acestea făcând parte din standardul Spice.

2.1 Definiția proiectului și fazele acestuia.

Managementul de proiect este considerat de mulți cercetători un domeniu relativ nou, aplicarea metodologiilor managementului de proiect a început între anii 1950 și 1960. Conform H. Kerzner (2006) [73] singura „diferență dintre managementul de proiect aplicat acum 40 ani și cel actual este răspândirea acestuia în întreaga structură a companiilor".

2.1.1 Istoria managementului de proiect

Domeniul managementului de proiect își are rădăcinile în antichitate, chiar dacă metodologiile folosite nu au fost documentate. Exemple de proiecte complexe sunt:

- construcția piramidelor [51];
- construcția marelui zid chinezesc [51];
- organizarea armatei romane[61];

Abia la începutul și mijlocul secolului 20 au apărut prima dată metodologii și organizații de management de proiect cunoscute și utilizate astăzi [57].

- diagrama Gantt;
- PERT (tehnica de evaluare și revizuire de program);
- teoria drumului critic inventată de Dupont Corporation;
- în anul 1965 a apărut asociația internațională a managementului de proiect;
- în anul 1969 a apărut Project Management Institute (PMI) cu scopul de a face din managementul de proiect o profesie;
- metoda PROMTII creată de Simpact Systems Limited se dorește a fi un ghid de dezvoltare a proiectelor software;
- mitul resursă lunară (man-month) descris de Fred Brooks. Acesta a descris pentru prima dată impactul negativ al adăugării resurselor într-un proiect

- aflat într-o stare avansată de dezvoltare; teoria constrângerilor descrisă de E. Goldratt în „The Goal";
- în anul 1986 metodologia SCRUM a fost definită ca un stil de management de proiect;
- în anul 1987 a fost publicată pentru prima dată „Project Management Body of Knowledge" de către PMI (Project Management Institute);
- între anii 1980 și 1990 a apărut pentru prima dată conceptul de valoare adăugată ca metodologie de management de proiect;
- în anul 1989 a apărut pentru prima dată metodologia PRINCE;
- în anul 1996 a apărut a două versiune a PRINCE, PRINCE 2;
- în anul 1997 E. Goldratt a descris metoda lanțului critic, metodă bazată pe teoria constrângerilor a aceluiași autor;
- în anul 1998 cartea „Project Management Body of Knowledge" a devenit un standard;
- în anul 2008 a apărut a patra versiune a „Project Management Body of Knowledge";
- în anul 2009 metodologia PRINCE2 a fost revizuită;
- în anul 2012 a apărut standardul ISO 21500:2012 care este un ghid a managementului de proiect;

2.1.2 Definirea proiectului

Există mai multe abordări a managementului de proiect, cele mai cunoscute sunt:

- ICB (International Competence Baseline) de la IPMA (International Project Management Association)[52];
- PRINCE 2 de la OGC (Office of Goverment Commerce) [63];
- Project Management Body of Knowledge de la PMI (Project Management Institute) [102];

Conform ICB [52] proiectul este „o operație constrânsă de timp și cost cu scopul de a realiza un set de livrabile la un anumit standard de calitate și cerințe definite anterior". Caracteristicile principale ale proiectului definite de ICB [52] sunt:

- standardul de calitate: realizarea obiectivului proiectului trebuie să respecte standardele de calitate impuse la startul proiectului;
- cerințe: îndeplinirea obiectivelor proiectului nu poate fi realizată fără a implementa cerințele sistemului definite la startul proiectului;

Conform PRINCE2 [63] proiectul se definește ca fiind „o organizație care este creată cu scopul de a livra unul sau mai multe produse conform unui plan de afaceri stabilit în prealabil". Particularitatea definiției proiectului conform PRINCE 2 [63] sunt:

- definiția organizației ca fiind compusă din membrii echipelor de dezvoltare a proiectului;
- livrarea produselor conform unui plan de afaceri presupune pentru prima dată existența unui contract semnat între furnizorul și clientul proiectului;

În viziunea PMI [102], proiectul reprezintă „un efort temporar care duce la realizarea unui produs unic, servicii sau rezultate". Fundamental pentru definiția dată de PMI este:

- factorul temporal: începutul și timpul de desfășurare a proiectului este bine definit;
- unicitate: rezultatul proiectului este unic, cuantificabil;
- obiectiv bine definit, sub forma unui produs, serviciu sau a unui rezultat;

2.1.3 Fazele dezvoltării proiectului

Managementul de proiect reprezintă totalitatea cunoștințelor, a tehnicilor și instrumentelor utilizate în realizarea obiectivelor proiectului [45]. Realizarea proiectului presupune implementarea unui ciclu de viață. Ca exemplificare, utilizata ulterior în teza, este ciclul de viață „V", specific proiectelor de dezvoltare software de tip automotive. Acesta este compus din următoarele faze:

- realizarea studiului de fezabilitate;
- realizarea caietului de sarcini și trimiterea acestuia către furnizor;
- specificarea arhitecturii sistemului;
- specificarea modulelor sistemului și implementarea acestora;
- testarea modulelor sistemului și integrarea acestora;
- testarea sistemului;
- - testarea de acceptanta și livrarea produsului;

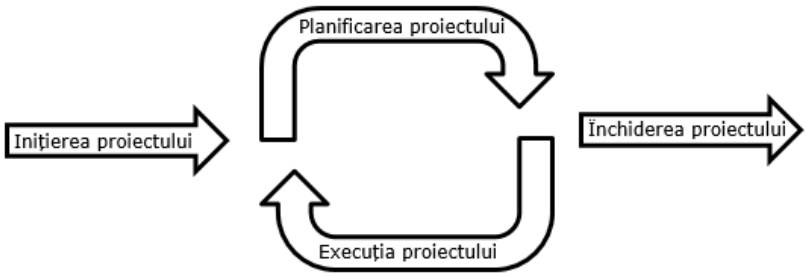

Fig. 2. 1 Fazele proiectului. Adaptare după Project Management Body of Knowledge [102].

Conform metodologiei de management de proiect [112], proiectul este compus din următoarele 4 faze:

- inițierea proiectului [59];
- planificarea proiectului[59];
- execuția proiectului[46];
- închiderea proiectului[46];

2.1.3.1 Inițierea proiectului: reprezintă procesul care facilitează autorizarea formală a unui nou proiect [102] (Ten Step PB, 2005). Înainte de a începe aceasta fază, următoarele activități trebuie să fie realizate [102] (Ten Step PB, 2005):

- documentarea ideii de afaceri și a cerințelor;
- realizarea studiului de fezabilitate;
- descrierea obiectivelor proiectului;

- definirea relației între proiect și organizație;descrierea conținutului proiectului, a livrabilelor, durata proiectului și prognoza asupra resurselor umane necesare;
- stabilirea managerului de proiect și a echipei de proiect;

2.1.3.2 Planificarea proiectului: are rolul de a planifica și coordona cu succes un proiect a unei organizații. În această fază se dezvoltă planul managementului de proiect prin identificarea, definirea și maturizarea conținutului proiectului, costului proiectului finalizându-se planificarea activităților proiectului. [102] (Ten Step PB, 2005).

Faza de planificare a proiectului este compusă din următoarele procese principale [102] (Ten Step PB, 2005):

- dezvoltarea planului de proiect,
- planificarea conținutului proiectului;
- definirea activităților și a secvențelor acestora;
- estimarea resurselor necesare proiectului;
- estimarea duratei activităților;
- planificarea implementării proiectului;
- estimarea și bugetarea costurilor;
- planificarea calității;
- planificarea resurselor umane;
- planificarea comunicării;
- planificarea managementului de risc;
- identificarea riscului;
- analiza calitativă a riscului;
- analiza cantitativă a riscului;
- planificarea reacțiunii la risc;
- planul de achiziții;
- planificarea încheierii contractelor cu furnizorii;

2.1.3.3 Execuția proiectului: conține procesele necesare realizării activităților descrise în planul managementului de proiect. Faza execuției proiectului conține următoarele procese [102] (Ten Step PB, 2005):

- coordonarea execuției proiectului;
- îndeplinirea criteriilor de calitate;
- achiziția echipei de proiect;
- dezvoltarea echipei de proiect;
- distribuția informației;
- solicitarea ofertelor furnizorilor;
- selectarea furnizorilor;

2.1.3.4 Închiderea proiectului: reprezintă rularea proceselor necesare terminării activităților proiectului. Cele două procese principale sunt [102] (Ten Step PB, 2005):

- închiderea proiectului;
- închiderea contractului;

2.2 Aspecte generale și particulare în realizarea proiectelor de înaltă tehnologie

Necesitatea de a răspunde cerințelor externe proiectelor a impus introducerea monitorizării, adaptării și optimizării ciclurilor de viață a proiectelor. Cerințele externe sunt definite ca fiind cerințele clienților (Dulce Domingos et al., 2013 [26]).

Nevoia definirii factorilor critici a proiectelor rezultă din existența diferitelor păreri asupra definițiilor și factorilor de succes a proiectelor:

- „direcționarea resurselor spre îndeplinirea unei sarcini unice încadrate într-un anumit timp, cost și calitate"(Olsen, 1971) [97]);
- „realizarea unui obiectiv specific, care implică o serie de activități și sarcini care consumă resurse" (Munns și Bjeirmi, 1996 [95]);
- „proiectul este un efort temporar care duce la realizarea unui produs unic, servicii sau rezultate" (PMBOK, 2005 [102]);

2.2.1 Aspecte generale ale proiectelor. Factori critici.

Dorința managerilor de proiect de a coordona proiecte conform teoriei managementului de proiect, duce deseori la neglijarea factorilor care pot influența desfășurarea proiectului.

De-a lungul timpului s-au identificat procese și factori noi care duc la îndeplinirea scopului proiectului. Neimplicarea acestor factori în urmărirea obiectivelor proiectului ar duce la un proiect fără manevrabilitate asemenea unei nave fără cârmă în largul unei furtuni.

Înființarea rețelelor de management de proiect au avut rolul de a identifica cele mai bune practici și de a îmbunătăți performanțele proiectului(Cooke-Davis, 2002 [22]).

Factorii critici ai succesului pot fi descriși ca variabile, caracteristici sau condiții care pot avea un impact semnificativ asupra succesului proiectului când acesta este susținut, întreținut sau coordonat adecvat (Leidecker JK, Bruno AV, 1984, Milosevic D., Patanakul P., 2005)[81][93]). În plus eficacitatea unei companii depinde în parte de succesul proiectelor sale (Kerzner H. 2000 [74], Cooper RG.2001 [23], Milosevic D., Patanakul P., 2005 [93]).

Proporția acestor factori depinde de domeniul și scopul proiectului. Spre exemplu:

- Kerzner (2006) [73] afirma că, clienții definesc calitatea necesară pentru acceptarea produsului. Dimensiunea succesului proiectelor este determinat de numărul de obiective și de definiții ale factorilor care duc la satisfacerea cerințelor clienților. Dimensiunea și complexitatea proiectului sunt criteriile care definesc numărul de obiective și factori.

- Reiss (1993)[103] definește proiectul incluzând resursa umană ca factor. Acesta sugerează că proiectul este o activitate umană care atinge un obiectiv clar, măsurat pe o scală a timpului. Pentru a

- realiza acest lucru, sugerează că managementul de proiect este o combinație de management și planificare și de gestionare a schimbării.

- Agarwal și Rathod (2006)[4] sunt de părere că atât clienții cât și echipa de dezvoltare sunt de acord că livrarea produsului cerut este cel mai important obiectiv. Dacă acest obiectiv nu este atins, atunci proiectul a fost un eșec.

- Kerzner(2006)[73] alegea calitatea ca fiind obiectivul care duce la îndeplinirea proiectului.

- Collins și Baccarini (2004)[21] au considerat că un conținut clar este un factor necesar care duce la îndeplinirea nevoilor clienților, automat ducând la succesul proiectului.

- Clarke (1999)[17] afirma că fără a defini clar conținutul proiectului, obiectivele dezvoltării pot fi vagi, iar echipa de dezvoltare poate pierde din vedere ceea ce trebuie să dezvolte.

- Dvir, D. Lipovetsky, S. Shenhar, A. & Tishler, A. [27] considerau următorii factori esențiali în atingerea obiectivelor proiectului:

  - Timp;
  - Cost;
  - Satisfacția clientului;

- Rose [104]; Hughes, S. W. Tippett, D. D. & Thomas, W. K. [65]; Dvir, D. Raz, T. & Shenhar, A. [28] ; Lim, C. S. & Mohamed, M. Z.[82]; Linberg, K. R.[83]; Munns, A.K. & Bjeirmi, B.F. [95] consideră următorii factorii decisivi în atingerea obiectivelor proiectului:

  - Timp;
  - Cost;
  - Calitate;

În Tabelul 2.1 se prezintă factorii cei mai răspândiți în proiecte și părerea mai multor cercetători asupra importanței acestora în proiecte.

Tabel 2. 1: Studii asupra factorilor de succes a proiectelor. Adaptat după [94].

Cercetători	Timp	Cost	Calitatea	Satisfacția clientului	Conținut
Dvir, D. Lipovetsky, S. Shenhar, A. & Tishler, A. [27]	■	■		■	
Rose, K. [104]; Hughes, S. W. Tippett, D. D. & Thomas, W. K. [65]; Dvir, D. Raz, T. & Shenhar, A. [28]; Lim, C. S. & Mohamed, M. Z. [82]; Linberg, K. R. [83]; Munns, A.K. & Bjeirmi, B.F. [95]	■	■	■		
Thomas, G. & Fernandez, W. [109]; Kerzner, H. [74]; Collins, A. & Baccarini, D.[21]; Belout, A. & Gauvreau, C. [12]; Armstrong, S. [7]	■	■	■	■	
Cohn și Ford [20]; Lindvall et al.[84]; Anda et al. [9]; Atkinson [8]; El Emam și Koru [70]; Kappelman et al. [77]; Lai [78]; Sumner et al. [86]; Yeo [72];	■	■	■		■

- Thomas, G. & Fernandez, W.[109]; Kerzner, H. [74]; Collins, A. & Baccarini, D. [21]; Belout, A. & Gauvreau, C. [12]; Armstrong, S. [7] consideră următorii factori ca fiind substanțiali în realizarea proiectului:

  - Timp;
  - Cost;
  - Calitate;
  - Satisfacția clientului;

- Cohn și Ford [20]; Lindvall et al.[84]; Anda et al [9]; Atkinson [8]; El Emam și Koru [70]; Kappelman et al [77]; Lai [78]; Sumner et al [86]; Yeo [72] consideră următorii factori ca fiind cei mai importanți în realizarea proiectului:

  - Timp;
  - Cost;
  - Calitatea;
  - Conținut;

2.2.2 Aspecte particulare în realizarea proiectelor automotive. Factori critici.

Succesul proiectelor în domeniul sistemelor de radio-navigație este dependent în mare măsură de factorii adoptați în urmărirea progresului proiectului. Nevoia de produse tehnologice avansate și lansarea acestora într-un timp cât mai scurt pe piață a devenit un obiectiv primordial în existența companiilor. Totodată se observă o legătură între factorii și obiectivele care duc la succesul proiectelor și implicit la succesul companiilor. Neimplicarea factorilor esențiali fiecărui domeniu de activitate poate duce la eșecul proiectelor.

Analizele efectuate au ca punct de pornire obiectivele clasice ale proiectelor, timp, cost și calitate, precum și factorii general valabili aplicabili în proiectele automotive.

Belassi și Tukel (1996)[11] au scris despre studiul realizat de Tukel și Rom (1967) în care managerii de proiect au fost rugați să identifice factorii critici pentru finalizarea cu succes a proiectelor lor.

Următorii șase factori au fost incluși în studiu:

- Suportul acordat de către management;
- consultarea clientului (comunicarea client-furnizor). Importanța consultărilor client-furnizor, precum și necesitatea învățării, predării și distribuirii abilităților de comunicare în structurile companiilor au fost descrise de către Badea, Proștean, Huțanu, Popa [1];
- estimările preliminare;
- disponibilitatea resurselor;
- performanța managerilor de proiect;
- alți factori;

Conform White, D. & Fortune, J. (2002)[113] cei trei factori critici în succesul proiectelor sunt:

- obiective clare;
- sprijinul managementului superior (senior management);
- fonduri și resurse adecvate;

Gabriela Fernandes et al. (2013) [35] considera ca îmbunătățirea cadrului proiectului este unul din obiectivele principale, acesta putând duce la succesul proiectelor.

Brown și Eisenhardt(1997)[14] au sugerat că factorii critici ai proiectului depind de gradul de standardizare a practicilor proiectelor.

Din experiența proiectelor automotive factorii de succes rezultă din structura proiectului. O structură de proiect bazată pe acțiuni spontane nu vor putea decât prin șansă să atingă obiectivele proiectului.

Analizând structura proiectelor de radio-navigație și modul de desfășurare a acestora, trebuie specificata importanța menținerii etapelor și a termenele stabilite inițial. Întârzierea cauzată de o componentă a sistemului poate duce la amânarea lansării unei noi mașini pe piață. Schimbarea caietului de sarcini în timpul implementării cerințelor inițiale este unul din principalele motive care duce la amânarea termenelor proiectului. De Bakker et al.(2000) [69] afirma că cerințele proiectelor software se vor schimba aproape sigur, iar acestea vor influența programul și costul proiectului.

Conținutul bine definit poate duce la o situație clară asupra caracteristicilor rezultatului proiectului, nu și la actualitatea acestuia. De exemplu în domeniul proiectelor de dezvoltare a telefoanelor mobile, cerințele clar definite nu garantează actualitatea produsului final. Acest lucru se datorează timpului de dezvoltare foarte redus, aproximativ șase luni, într-un domeniu în care actualitatea produsului este foarte importantă.

Deoarece în domeniul proiectelor sistemelor de radio-navigație, cerințele tehnologice se pot schimba, coordonatorii de proiect sunt nevoiți să improvizeze pentru a îndeplini scopul proiectului.

Conform Wateridge (1995) [111] sponsorul sau clientul proiectului ar trebui să identifice criteriile și factorii de succes relevanți în atingerea obiectivelor proiectelor. În acest context clientul sistemelor de radio-navigație este întotdeauna interesat să implementeze cele mai noi tehnologii deoarece progresul tehnologic în acest domeniu este foarte rapid (Huțanu et al. 2013 [5]), îndeplinind cerințele factorului „actualitatea produsului”.

Se observă din studiu că domeniul proiectului influențează criteriile care trebuie urmărite în realizarea obiectivelor. Chiar dacă unii cercetători consideră că proiectele vor avea succes numai luând în calcul toate criteriile cunoscute, acest lucru nu a putut fi confirmat de practică. Factorul esențial în domeniul electronic și automotive, actualitatea produsului, nu a fost luat în calcul.

Importanța resursei umane și a proceselor automotive Spice (Software Process Improvement and Capability Determination) în proiecte a fost subliniată de Reiss(1993)[103] respectiv Hoermann et al. (2008)[76].

Spice este o normă a standardului ISO/IEC 15504. Acest standard a fost publicat în anul 1998 în urma mai multor ani de studiu. Standardul Spice ajută la verificarea nivelelor proceselor de implementare în companii.

În automotive, standardul Spice a fost adoptat în anul 2001 și are ca scop verificarea eficienței proceselor folosite în dezvoltarea proiectelor informatice. La realizarea standardului au luat parte cei mai importanți producători de automobile ca Audi, BMW, Daimler, Fiat, Ford, Jaguar, Land Rover, Porsche, Volkswagen, Volvo [44].

În funcție de gradul de maturitate al unei structuri sau proiect, aceasta poate avea următoarele grade de maturitate [76]:

- nivelul 0: procese incomplete, acțiuni spontane;

- nivelul 1: procese intuitive dar nestructurate;
- nivelul 2: procese controlate și structurate;
- nivelul 3: procese consacrate;
- nivelul 4: procese previzibile;
- nivelul 5: procese de optimizare, control total;

Gradele de maturitate sunt definite în funcție de nouă atribute de proces (Fig. 2.2).

În urma auditării proceselor conform standardului Spice [76], se identifică gradul de existență și punere în practică a proceselor. Scala de notare a celor doi factori este:

- neîndeplinit, dacă gradul de realizare este între 0% și 15%;
- parțial îndeplinit, dacă gradul de realizare este între 16% și 50%,
- în mare măsură îndeplinit, dacă gradul de realizare este între 51% și 85%;
- în totalitate îndeplinit, dacă gradul de realizare este între 86% și 100%;

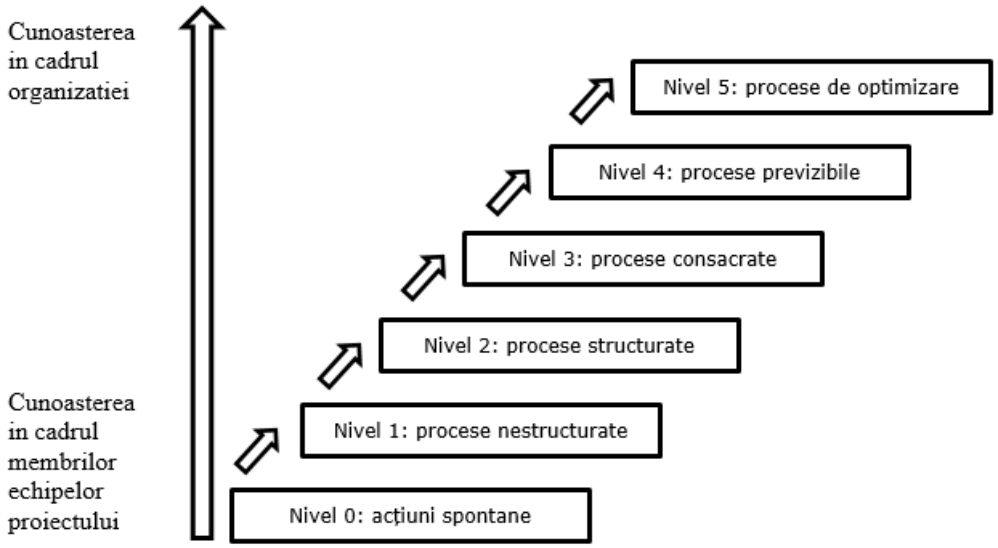

Fig. 2. 2 Treptele gradelor de maturitate conform SPICE [76].

Reiss [103] definește importanța resurselor umane în organizarea și desfășurarea proiectelor, Hoermann [76] la rândul său descrie o serie de procese necesare în domeniul dezvoltării de sisteme de radio-navigație.

Procesele și factorii principali necesari dezvoltării proiectelor automotive sunt[76] :

- Monitorizarea furnizorului:

Obiectivul procesului de monitorizare a furnizorilor este de a verifica performanța proceselor furnizorului astfel încât proiectul să îndeplinească cerințele convenite. „Scopul procesului de monitorizare a furnizorului este de a monitoriza performanța furnizorului în comparație cu cerințele proiectului" Hoermann et al. (2008)[76].

Principala activitate a acestui proces constă în menținerea comunicării la toate nivelele, de la departamentul de testare până la managementul proiectului. Foarte important în acest sens este identificarea interfețelor de comunicare cât și recunoașterea acestora. Practicile comune pentru a îndeplini procesul de monitorizare a furnizorului sunt:

- identificarea interfețelor de comunicare;
- schimbul de informații;
- revizuirea dezvoltării tehnice a furnizorului;
- revizuirea progresului proiectului;
- urmărirea problemelor nerezolvate;
- acțiunea de corectare a erorilor;
- acceptarea cererilor de modificare;

Acțiunea de corectare a erorilor trebuie să fie urmărită îndeaproape, deoarece de cele mai multe ori din cauza constrângerilor de timp este necesara stabilirea priorității de înlăturare a erorilor. Aceasta va duce la întâlniri foarte dese, scopul acestor ședințe fiind de a rezolva problemele stringente ale proiectului. Dacă definirea priorității erorilor nu are loc din timp, există riscul ca livrabilul proiectului să conțină erori majore care nu mai pot fi rezolvate la timp.

- Managementul lansării produselor

„Scopul procesului de livrare a produselor este acela de a controla livrabilele către client" Hoermann et al. (2008)[76]. Se înțelege prin livrabile, componentele software și hardware care compun sistemul. Livrarea stadiilor intermediare ale proiectelor automotive face parte din înțelegerile inițiale între clientul și furnizorul proiectului. Managementul lansării produselor presupune planificarea fiecărei livrări intermediare cât și conținutul acesteia.

- Analiza cerințelor

„Scopul acestui proces este acela de transforma cerințele clienților în cerințe tehnice care vor duce către modelarea sistemului" Hoermann et al. (2008)[76]. Acest proces presupune ca cerințele să fie clar exprimate. Orice cerință incomplet definită va genera mai târziu costuri de modificare a caietului de sarcini.

- Integrarea sistemului și testarea acestuia

„Scopul procesului de testare a integrării de sistem este acela de a integra componentele sistemului, de a produce un sistem integrat care va corespunde arhitecturii sistemului și a cerințelor clientului definite în cerințele de sistem". Hoermann et al. (2008) [76].

Conceptul integrării și testării unui sistem presupune implementarea strategiilor de integrare și testare a sistemului. Trebuie căutate răspunsuri asupra modalităților de integrare a diferitelor componente ale sistemului cât și a modalităților de testare a sistemului.
Dificultățile pot apărea în momentul în care testarea unui sistem presupune interconectarea cu un alt sistem cu un grad de maturitate inferior.
Principalul scop al acestui proces este găsirea unor soluții de simulare care să ducă la îndeplinirea sarcinilor procesului.

- Managementul de proiect
„Scopul acestui proces este de a identifica, planifica, coordona și monitoriza activitățile și resursele necesare unui proiect pentru a crea un produs în contextul cerințelor proiectului și a constrângerilor" Hoermann et al. (2008)[76]. Inexistenta unui management de proiect adecvat ar putea duce la realizarea unui produs neconform sau la nerespectarea termenului limită de predare a sistemului.
- Managementul riscului
„Rolul acestui proces este de a identifica, analiza, trata și monitoriza în mod continuu riscul." Hoermann et al. (2008)[76].

Netratarea riscului unui proiect poate duce la eșecul acestuia. Exemple de riscuri care pot apărea în proiect sunt limitarea resurselor, lipsa echipamentelor de testare, decizii manageriale târzii etc.

Concluzionând, în domeniul automotive și cel al tehnologiilor de vârf, neimplicarea factorului tehnologic și al timpului va duce la neglijarea importanței de a dezvolta și lansa produse noi pe piață. De exemplu neintegrarea protocoalelor de comunicare a celor mai noi telefoane mobile în sistemele automotive ar putea duce la eșecul proiectului. Analizând implicațiile acestui exemplu, se observă că această decizie ar putea duce la pierderea unui număr semnificativ de clienți. Aceasta deoarece tot mai mulți clienți își doresc utilizarea interfeței telefonului mobil în sistemele de radio-navigație.

2.3 Scopul proiectelor.

Realizarea produselor unice de înaltă tehnologie respectând factorii de timp, cost, calitate și funcționalitate reprezintă scopul principal a proiectelor de radio-navigație (adaptare după [102]). Prin realizarea de produse unice și de înaltă tehnologie, companiile dezvoltatoare își pot asigura un avantaj competițional important în comparație cu celelalte firme concurente. Factorii timp și calitate sunt acei factori care vor determina în mare măsură piața să accepte un produs ca fiind „numărul 1" în domeniu.

Calitatea și funcționalitatea sunt acei factori care determină utilizatorul final să cumpere un anumit produs. Totuși din exemplul telefoanelor inteligente (smartphone), în momentul în care factorul timp este pe planul al doilea, succesul proiectului are de suferit. Spre exemplu unul din primele telefoane inteligente fabricate de Apple, care a respectat factorul timp, cost, calitate și funcționalitate a adus succes companiei. Acest lucru se datorează lansării în momentul optim al produsului pe piață. Prin această strategie, compania a reușit să devină un etalon în domeniul telefoniei inteligente.

Același lucru este valabil și în alte domenii incluzându-l pe cel automotive. De exemplu întârzierea lansării unei noi funcții inovatoare va duce la potențiala pierdere a clienților, lansarea întârziată cu un an putând duce la pierderea caracteristicii de etalon în domeniu.

Respectarea factorului timp în proiecte cât și dezvoltarea de tehnologii inovatoare este principalul scop al proiectelor. Îndeplinind acești factori, proiectul va duce automat la câștigarea cotei de piață și la creșterea profitului companiei.

Principala problemă în realizarea produselor este diferența de opinie dintre sponsorul proiectului și dezvoltatorul acestuia. Din dorința de a implementa cele mai

noi tehnologii, sponsorul va aduce schimbări cerințelor proiectelor până în fazele târzii ale acestora. Aceste schimbări pot aduce modificări calendarului proiectului, fapt ce va intra în coliziune cu dorința furnizorului de a finaliza proiectul în termenii stabiliți inițial.

Definind scopul proiectului ca fiind realizarea unui produs unic în termenii temporali stabiliți inițial, schimbarea cerințelor în fazele avansate ale proiectului cresc riscul neîndeplinirii obiectivelor proiectelor. Riscul la rândul lui trebuie tratat astfel încât să nu se concretizeze. Implementarea proceselor de analiză a schimbărilor este doar una din masurile care sunt luate de către companii.

Realizarea proiectelor respectând factorul temporal este una din cele mai mari provocări. Implementarea de măsuri care să ducă la îndeplinirea obiectivelor proiectelor este una din sarcinile principale ale managerilor de proiect.

2.4 Concluzii.

Capitolul de față sistematizează definirea complexă a managementului de proiect, pornind de la istoria acestuia și până la particularizări ale proiectelor de dezvoltare software pentru sistemele de radio-navigație. Contextul de dezvoltare a fiecărui tip de proiect a atras după sine dezvoltarea tehnicilor diversificate ale managementului proiectelor.

Analiza definițiilor proiectelor și sinteza fazelor proiectului reprezintă o înșiruire cronologică normală care stă la baza studiului științific.

Analiza și descrierea diferitelor aspecte ale proiectului au condus la următoarele concluzii:

- atingerea obiectivelor proiectelor presupune cunoașterea detaliată a fazelor, pașilor proiectelor, și a ordinii în care aceștia trebuie să se desfășoare, precum și a factorilor critici specifici tipului de proiect;

- înțelegerea importanței factorilor critici stă la baza succesului proiectului, în domeniul automotive acest lucru fiind posibil având în vedere faptul că literatura de specialitate abordează profund aceste aspecte;

- standardizarea proceselor utilizate în proiectele automotive, sub forma proceselor Spice, ajută coordonatorii de proiect în organizarea și luarea deciziilor pentru metodele de control și comandă a acestor tipuri de proiecte;

Contribuțiile personale ale autorului sunt:

- sinteza fazelor reprezentative proiectelor în general cu particularizări în proiectele automotive;

- sinteza factorilor critici reprezentativi proiectelor pe baza unei analize comparative din literatura de specialitate;

- descrierea factorilor critici reprezentativi proiectele automotive precum și prezentarea factorilor critici specifici proiectelor automotive. Acești factori stau la baza studiului și sunt:

- actualitatea produsului;

- factorului temporal;

- identificarea unui nou factor critic reprezentativ proiectelor automotive, mai concret acela al produsului inovativ, care începe să aibă o pondere prioritară ca factor critic față de cei considerați până acum ca reper absolut, timp, cost și calitate.

3. CICLURILE DE VIAŢĂ ALE PROIECTELOR

Capitolul de față este organizat în patru părți. Prima parte prezintă caracteristicile generale ale ciclurilor de viață ale proiectelor de dezvoltare software.

A două parte descrie aspectele generale ale modelului de dezvoltare tradițional în „V".

Partea a treia analizează principalele metodologii AGILE, respectiv Adaptive Software Development (ASD), Crystal, Feature Driven Development (FDD), Agile Model Driven Development (AMDD), SCRUM. Dynamic System Development (DSDM), Lean Development și Extreme Programming.

Partea a patra concretizează obiectivul principal al acestui capitol, mai concret, acela de scoatere în evidentă în mod detaliat a aspectelor particulare ale modelelor de dezvoltare a proiectelor automotive cât și efectele asupra calendarului proiectului în cazul implementării modificării cerințelor în diferitele faze ale proiectului.

3.1 Cicluri de viață. Aspecte generale.

Ciclurile de viață reprezintă în fapt o înlănțuire de procese care îndrumă proiectul de la începutul acestuia până la finalizarea lui. Conform PMBOK [102] ciclurile de viață ale proiectelor definesc fazele care leagă începutul și sfârșitul proiectului, iar [54] descrie ciclul de viață al unui proiect ca fiind definiția fazelor intermediare a unui proiect, program sau portofoliu și care oferă o structură de control a progresului proiectului.

Cel mai des întâlnite modele de cicluri de viață pentru proiecte de dezvoltare software sunt:

- **modelul în V:** este un mod de a reprezenta proiectarea unui sistem precum și a procesului de dezvoltare [48];

- **metodologiile AGILE:** sunt metode de dezvoltare software flexibile, rapide, receptive la schimbări[55];
    - Indiferent dacă este vorba despre cicluri de viață în V sau cicluri de viață AGILE, acestea se compun în general din următoarele faze:

- **definirea cerințelor**: reprezintă funcționalitățile și caracteristicile produsului final care urmează a fi implementate de-a lungul dezvoltării proiectului;

- **definirea arhitecturii proiectului**, inclusiv a programelor software: reprezintă structura modulară și funcțională care determină modalitatea de comunicare din cadrul sistemului;

- **implementarea tuturor modulelor sistemului**: în această fază a proiectului se implementează(codarea programului) toate cerințele definite în faza de definire a proiectului;

- **testarea modulelor sistemului**: reprezintă testarea modulelor în mod individual, fără a lua în calcul interfețele de comunicare cu celelalte module componente ale sistemului;

- **integrarea modulelor într-un sistem unic**: reprezintă integrarea modulelor proiectului astfel încât ansamblul final să furnizeze funcționalitățile programului definit de cerințele inițiale;

- **testarea sistemului**: reprezintă o fază avansată de testare prin intermediul căreia se verifică dacă sistemul furnizează toate funcționalitățile definite, apelând la toate variantele, setările și codificările incluse în sistemul de comunicare.

- **mentenanța proiectului**: în urma lansării produsului, acesta va fi predat echipei de întreținere a produsului. Dacă în timpul folosirii produselor pe piață apar diverse probleme, este de datoria echipei de întreținere să raporteze și să evalueze gravitatea erorilor. În funcție de rezultatul acestei analize, echipa de întreținere decide reducerea abaterilor prin module de ajustare ale software-ului existent.

Procesele și metodologiile utilizate în dezvoltarea proiectelor software, diferă de la proiect la proiect. Poppendieck afirma într-o scrisoare către Mnkandla(2008)[33] că folosirea metodelor clasice de dezvoltare își au din ce în ce mai puțin rostul. Dezvoltarea de tehnologii noi va duce în mod direct și la adaptarea proceselor de dezvoltare utilizate.

Folosirea diverselor metodologii în proiectele de dezvoltare software a dus la împărțirea opiniilor cercetătorilor. Conform Larmann(2004)[79] și Martin (1999)[90] metodologiile iterative se pot aplica adecvat în proiecte mici. Chiar dacă utilizarea metodologiilor agile nu sunt suficiente dezvoltării proiectelor de mari dimensiuni (Turk et al. 2002[110] și Maurer et al. 2002[89]), la ora actuală metodele clasice necesită în permanență ajustări bazate pe metodologiile AGILE, datorită dinamicii schimbărilor apărute în cadrul proiectelor software.

Coordonatorii de proiect aleg modelul ciclurilor de viață a proiectelor în funcție de:

- relevanța metodologiei pentru proiect;
- adaptabilitatea metodologiei la cultura țării în care se desfășoară proiectul;
- costurile implementării metodologiei;
- experiența echipei de dezvoltare cu metodologia de lucru și necesitatea instruirii echipei de dezvoltare;

3.2 Modelul în V

Modelul V a fost conceput și dezvoltat ca urmare a nevoii de îmbunătățire a ciclului de viață în cascadă, fiind un model care prezintă în mare măsură aceleași faze propuse și de modelul în cascadă.
Fazele modelului în cascadă sunt [34]:

- identificarea și documentarea specificațiilor;
- proiectarea;
- implementarea;

- integrarea;
- testarea;
- instalarea software-ului pe sistemul hardware;
- mentenanța;

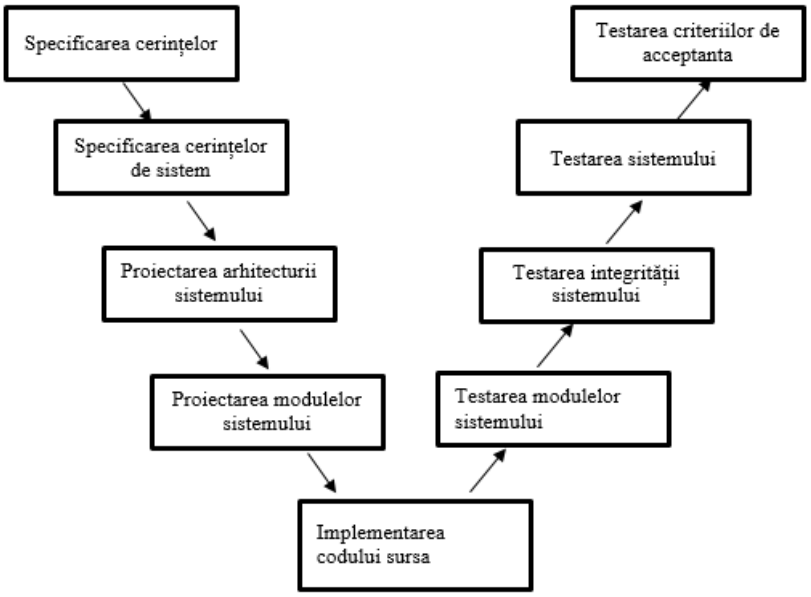

Fig. 3. 1 Modelul în V. Adaptare după [54].

O particularitate a modelului în V [Fig. 3. 1] este implementarea unor porți de calitate care să permită proiectului să treacă în faza următoare.

În comparație cu modelul în cascadă, modelul în V mai prezintă următoarele faze:

- Specificarea și documentarea cerințelor;
- Specificarea cerințelor de sistem;
- Proiectarea arhitecturii sistemului;
- Proiectarea modulelor sistemului;
- Implementarea codului sursă;
- Testarea modulelor sistemelor;
- Testarea integrității sistemului (teste de integrare);
- Testarea sistemului;
- Testarea criteriilor de acceptanță;

3.3 Modelul ADDIE

Modelul ADDIE este un model de dezvoltare utilizat foarte des în conceperea instrumentelor de învățare [53]. Fazele modelului sunt:

- Faza de analiză: mediul învățării, scopul și obiectivul sunt definite în cadrul acestei faze;

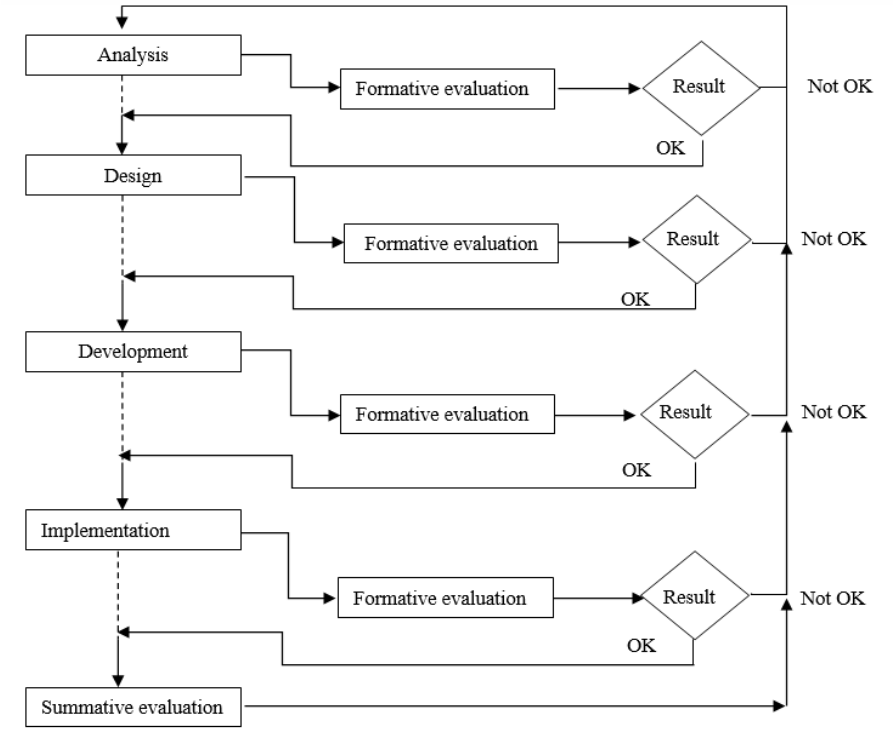

Fig. 3. 2 Modelul ADDIE [114].

- Faza de proiectare: presupune planificarea metodelor de învățare, exerciții și instrumente de învățare. Fiecare etapă a acestei faze trebuie să fie logică și trebuie parcursă cu atenție la detalii;
- Faza de dezvoltare: presupune faza în care programatorii scriu codul sursă;
- Faza de implementare: în această fază se implementează strategia de predare a cunoștințelor;
- Faza de evaluare: reprezintă măsurarea eficacității și a eficienței instrucțiunilor dezvoltate.

Modelul ADDIE (Fig. 3. 2 **Modelul ADDIE [114].** se aplică preponderent în proiecte de dezvoltare de înaltă tehnologie din domeniul învățării și a transferului cunoașterii. Aplicarea acestuia în proiectele de radio-navigație nu este posibilă din cauza:

- Timpului necesar adaptării acestuia la cerințele unui model aplicat în domeniul automotive;
- Inexistența posibilității definirii livrabilelor la intervale fixe de timp;
- Imposibilitatea conectării acestui model la procesele de producție;

3.4 Metodologiile AGILE

Metodologiile AGILE sunt utilizate primordial în dezvoltarea sistemelor IT. Termenul agile în proiectele de dezvoltare software a apărut ca urmare a unui

manifest în anul 2001 [42]. Metodologiile AGILE au luat naștere ca urmare a unei întâlniri la care au participat reprezentanți ai Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming. Aceste metodologii au fost create ca urmare a nevoii unei alternative la procesele de dezvoltare bazate pe documentații stricte.

Valorile metodologiilor AGILE sunt[42]:

- oamenii și interacțiunile lor sunt mai importanți și au prioritate înaintea proceselor și instrumentelor;
- obținerea funcționalității software are prioritate față de realizarea documentației cuprinzătoare;
- colaborarea cu clientul este mai importantă decât negocierea contractului;
- receptivitatea la schimbări are prioritate față de urmărirea unui plan strict;

Cele mai utilizate metodologii AGILE sunt:
- Adaptive Software Development (ASD);
- Crystal;
- Feature Driven Development (FDD);
- Agile Model Driven Development (AMDD);
- Scrum;
- Dynamic System Development Methodology (DSDM);
- Lean Development;
- Extreme Programming;

3.4.1 ASD (Adaptive Software Development)

ASD(Adaptive Software Development) este un model care a fost dezvoltat de către James A. Highsmith III și publicat de acesta în anul 2000[66]. Principiile acestui model se bazează în mare parte pe studiile efectuate de Highsmith legate de metodele iterative de dezvoltare. Acest proces descrie în principal modele de lucru pentru proiectele mari, ASD nepliindu-se pe ciclurile de viață ale proiectelor de mici dimensiuni.

Ciclul modelului ASD este compus din 3 faze [Fig3.3]:

- **Speculareа** : înlocuiește faza de planificare din procesele clasice. Deoarece proiectele au momente imprevizibile, coordonatorii de proiect trebuie să improvizeze și să speculeze situațiile. Planificarea unei situații neprevăzute nu poate fi realizată.

- **Colaborarea** : subliniază importanța muncii în echipă. Tehnica comunicării folosită este JAD (Joint Application Development), o întâlnire între furnizorul și sponsorul proiectului. Utilizarea acestei tehnici permite proiectului să răspundă rapid cerințelor proiectului.

- **Învățarea** : subliniază nevoia de învățare din greșeli. Faza de învățare presupune revizuirea fazelor anterioare și crearea de metode îmbunătățite. Astfel se asigură evitarea acelorași probleme într-o fază târzie a proiectelor.

Tehnica principală folosită de metodologia ASD este JAD(Joint Application Development). JAD este în fapt o întâlnire între clienți și programatori care are ca scop stabilirea funcțiilor sistemului și îmbunătățirea comunicării.

Caracteristicile ASD descrise de Highsmith în [66] sunt:

- toate activitățile proiectului trebuie să fie îndreptate către realizarea obiectivului proiectului;
- dezvoltarea nu trebuie să fie orientată spre îndeplinirea sarcinilor, ci pe dezvoltarea funcțională a software-ului;
- iterativ: mediul de dezvoltare este de cele mai multe ori turbulent, de aceea efortul trebuie orientat către „repararea" sistemului și nu către realizarea sistemului din prima încercare;

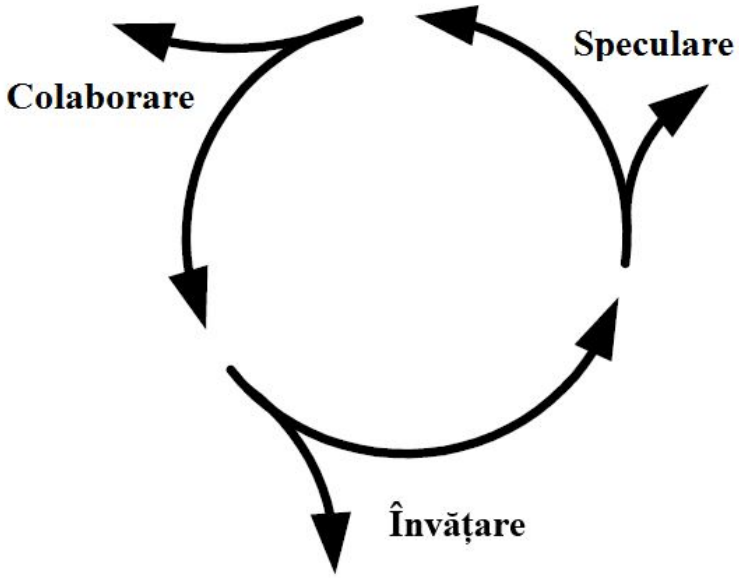

Fig. 3. 3 Ciclul ASD [66] [101]

- prin realizarea unor obiective intermediare tangibile se atenuează ambiguitățile din proiect, acestea ducând la decizii de compromise încă din faze timpurii în proiect;
- toleranța la schimbări: schimbările în proiectele software sunt dese. Acesta este motivul pentru care componentele sistemelor trebuie dezvoltate în așa fel încât să poată fi modificate cu un efort minim;
- reacțiune la risc: Implementarea componentelor cu risc mare trebuie să înceapă cât mai devreme posibil;

3.4.2 Crystal

Crystal include un număr de metodologii care vor fi folosite individual pentru fiecare proiect. Crystal propune alegerea metodologiei în funcție de dimensiunea și criticitatea proiectului.

Proiectele Crystal sunt caracterizate prin cicluri de dezvoltare incrementale. Aceste cicluri au o durată de maxim 4 luni, de preferat însă între 1 și 3 luni (A.

Cockburn, 2002) [18]. Metodologiile Crystal subliniază importanța comunicării în proiect. Totodată Crystal nu limitează adaptarea altor metodologii AGILE precum SCRUM sau XP (A. Cockburn, 2002) [18].

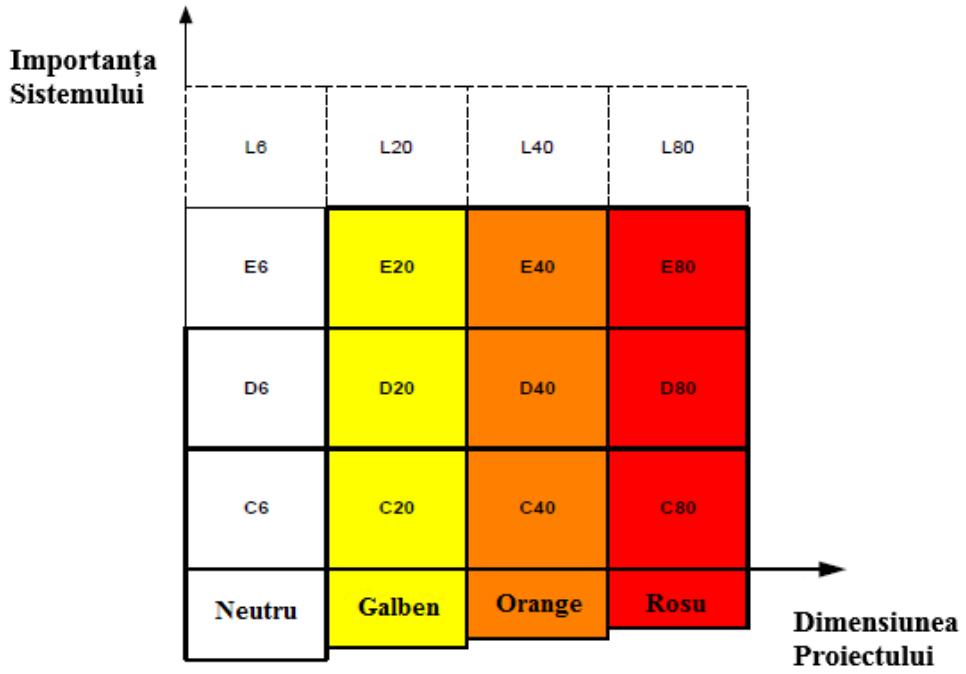

Fig. 3. 4 Dimensiunile metodologiilor Crystal [101] [18].

Principalele metodologii Crystal sunt:

- **Crystal Clear**: este o metodologie folosită în proiectele de mici dimensiuni, cuprinzând până la 6 programatori. Prin anumite extensii de comunicare și testare, aceste proiecte pot cuprinde până la maxim 10 componenți (A. Cockburn, 2002) [18];

- **Crystal Orange**: este folosită în proiectele de până la 40 de componenți și cu o durată a proiectului de până la 2 ani. Proiectele sunt împărțite în mai multe echipe funcționale, importantă fiind strategia „time-to market" (A. Cockburn, 1998) [19];

Similitudinile între cele două metodologii sunt:

- livrarea incrementală regulată a proiectelor;
- urmărirea evoluției proiectului bazată pe livrările componentelor software și a deciziilor majore în defavoarea documentației;
- implicarea directă a clientului;
- testarea automată a funcționalităților sistemului;
- organizarea de ședințe cu scopul de a îmbunătăți produsul și metodologia la începutul și mijlocul incrementului;

Singura diferență majoră a celor două metodologii o reprezintă livrarea rezultatelor intermediare la fiecare 2-3 luni în metodologia Crystal Clear și maxim 4 luni în metodologia Crystal Orange.

Tehnicile folosite în Crystal sunt:

- controlul versiunilor;
- programare;
- testare;
- comunicare;
- urmărirea proiectului;
- desenarea și măsurarea performanței;

3.4.3 FDD (Feature Driven Development)

Spre deosebire de alte procese, FDD (Feature Driven Development) nu acoperă întregul proces de dezvoltare software. Acesta se concentrează pe două faze:

- faza de proiectare;
- faza de implementare;

Asemenea metodologiilor Crystal, FDD se bazează pe principiul dezvoltării iterative și a celor mai bune practici pentru domeniul de dezvoltare a proiectului. Feature Driven Development a fost definit de Jef de Luca și a fost gândit pentru proiecte mari și critice.

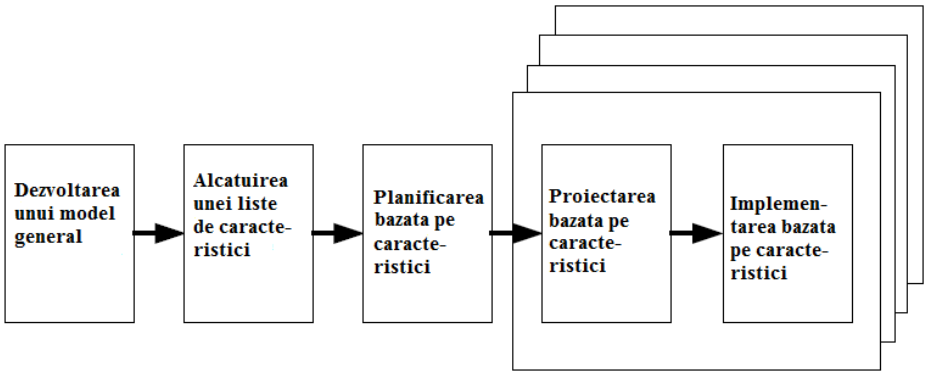

Fig. 3. 5 Procesul FDD [101][99]

FDD este compus din 5 procese iterative care se desfășoară de-a lungul a 2-3 săptămâni[101]:

- **dezvoltarea unui model general**: în această fază se presupune că, conținutul proiectului este cunoscut. În această fază nu se discută despre cerințe în detaliu, ci este mai degrabă o descriere generală a sistemului;

- **alcătuirea unei liste de caracteristici**: în această fază se descriu cerințele principale ale fiecărui modul, cerințe care în prealabil au fost negociate cu clientul proiectului;

- **planificarea bazată pe caracteristici**: planificarea se face în funcție de prioritatea fiecărei cerințe;

- **proiectarea bazată pe caracteristici**;

- **implementarea bazată pe caracteristici**: proiectarea și implementarea bazată pe caracteristici reprezintă împărțirea unui modul în subseturi și implementarea acestora în mod iterativ. La finalul implementării, modulele vor fi integrate și vor avea ca rezultat un modul funcțional al sistemului;

3.4.4 AMDD (Agile Model Driven Development)

Este o versiune a metodologiei de dezvoltare bazată pe modelare. Aceasta nu descrie un proces în sine ci combină mai multe metodologii, creând un cadru de modelare.

Acest model a fost dezvoltat de Scott Ambler în anul 2000. Inițial el s-a numit Extreme Modelling, însă la sugestia lui Robert Cecil Martin, acesta a primit denumirea de Agile Model în anul 2001.

Diferența față de alte modele constă în faptul că înainte de a începe implementarea codului propriu-zisa, nu se așteaptă crearea de modele de programare detaliate. Acestea se bazează pe modele cadru, care permit începerea codificării [43].

O alta caracteristică este alegerea modelului folosit. AMDD nu insistă pe un model anume. Mesajul care îl dă este necesitatea creării unui model înainte de a începe implementarea cerințelor software.

Comunicarea în cadrul echipei de dezvoltare joacă un rol foarte important. Acesta este și motivul pentru care dimensiunea echipei nu poate fi mare.

Un alt motiv pentru care acest model nu se poate aplica în echipe de mari dimensiuni sunt instrumentele simple folosite(de obicei se modelează pe hârtie).

Deoarece este un model relativ nou, în practică nu se cunosc cazuri în care acest model să fi fost folosit.

3.4.5 SCRUM

SCRUM a fost definit în 1986 de către Hirotaka Takeuchi și Ikujiro Nonaka ca fiind o strategie globală și flexibilă care permite unei echipe să îndeplinească un țel comun (H. Takeuchi și I. Nonaka, 1986) [40].

„Este un cadru în care se pot adresa probleme complexe și adaptive, în timp ce se livrează rezultate la același nivel de productivitate și calitate” [64].

SCRUM se dorește a fi o metodologie care să crească viteza și flexibilitatea de dezvoltare. Pentru a îndeplini acest scop, SCRUM folosește diferite variabile menite să fie modificate de-a lungul dezvoltării proiectelor.

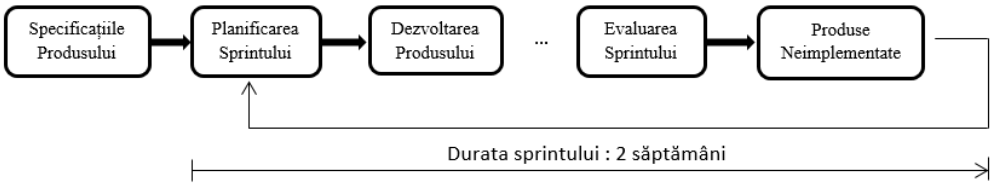

Fig. 3. 6 Reprezentarea grafică a SCRUM [2-Hutanu].

Practicile SCRUM se împart după cum urmează (K Schwaber, M Beedle, 2002) [71]:

- **caracteristicile produsului** sunt prezentate sub forma unor cazuri de utilizare: definește caracteristica produsului final bazându-se pe cerințele existente;

- **estimarea efortului** : este un proces iterativ în care elementele din lista de funcțiuni sunt estimate cu acuratețe mai mare, date fiind informațiile suplimentare avute la momentul estimării;

- **Sprint**: este faza incrementală de execuție a produsului, orice activitate fiind atribuită unui sprint [2- Hutanu];

- **ședința de planificare a Sprint-**ului este organizată în două faze. Una de definire a conținutului sprintului. Planificarea se face împreună cu clientul proiectului. În cea de-a două fază, accentul se pune pe modul de implementare a cerințelor;

- **caracteristicile ce urmează a fi** implementate : este punctul de pornire a fiecărui sprint și conține lista de cerințe ce urmează a fi implementate în sprintul curent;

- **ședințe zilnice de monitorizare** numite ședințe zilnice SCRUM: sunt întâlniri în care se stabilesc pașii următori ce trebuie realizați până la următoarea întâlnire;

- **recenzia sprintului finalizat**: este o ședință informală în care se prezintă stadiul dezvoltării;

Utilizarea tehnicilor SCRUM se pliază cel mai bine pe proiectele de dimensiuni medii și cu cerințe instabile, în timp ce echipele de dezvoltare sunt de dimensiuni reduse. Echipele de dezvoltare de dimensiuni reduse avantajează comunicarea în interiorul proiectului, fapt ce determină reacțiune rapidă la schimbarea cerințelor.

3.4.6 DSDM (Dynamic System Development Methodology)

Ca și celelalte metodologii agile, DSDM (Dynamic System Development Methodology) prezintă o dezvoltare incrementală. Metodologia a luat naștere în 1994 și are ca scop reducerea rebuturilor [101].

Ideologia DSDM se deosebește de celelalte metodologii prin faptul că la început sunt definite resursele umane necesare și calendarul proiectului. Într-un alt doilea pas se vor defini cerințele în așa fel încât resursele existente și calendarul proiectului să nu fie depășite.

Procesul metodologiei DSDM conține 5 faze:

- studiul de fezabilitate;
- crearea planului de afaceri;
- crearea modelului funcțional;
- proiectarea și construcția iterațiilor;
- implementarea;

Dimensiunile echipelor din proiecte poate varia între 2 și 6 persoane, însă în proiecte pot fi utilizate mai multe echipe. Acesta este motivul pentru care metodologia DSDM poate fi utilizată în proiecte de mari dimensiuni cât și de dimensiuni reduse.

Scopul metodologiei este de a răspunde „neajunsurilor" modelelor tradiționale de dezvoltare, în principal rigiditatea modelelor. DSDM se dorește a fi o metodologie de dezvoltare:

- rapidă;
- receptivă la schimbări;

DSDM poate fi adaptată astfel încât să poată fi integrată în cadrul altor metodologii sau poate fi un cadru „liant" între diferite metodologiile utilizate în proiecte de mari dimensiuni.

3.4.7 Lean Development

Acest model își are rădăcinile în procesul Lean Manufacturing, Lean Development fiind o translație în proiectele de dezvoltare software.

Filozofia procesului este de a crea software tolerant la schimbări cu o treime din:

- efortul uman;
- orele de dezvoltare;
- investițiile în unelte;
- și efortul de adaptare la noul mediu de piață;

Lean Development permite clienților să-și amâne cât de mult posibil deciziile privind cerințele. Totodată prioritatea asupra cerințelor implementate va fi dată de valoarea vizibilă pentru client.

Principiile metodologiei sunt [48]:

- **eliminarea rebuturilor**. Cauzele rebuturilor în cazul proiectelor software sunt:
  - implementarea inutilă de funcționalități;
  - demararea mai multor activități decât sunt necesare;
  - întârzierile din procesele de dezvoltare software;
  - cerințe neclare;
  - schimbarea cerințelor;
  - lipsa de comunicare;
  - lipsa testării;
  - problemele de calitate;
- **respectarea principiilor de calitate** prin:

- revizuirea codului sursă;
- testarea imediată a funcționalităților;
- revizuirea funcționalităților;
- comprimarea timpilor între etapele procesului;
- creșterea numărului de integrări ale modulelor sistemului;
- automatizarea testării;

- **crearea cunoașterii**: este un factor esențial în creșterea performanței. Extinderea cunoașterii la toate nivelurile proiectului va duce la înțelegerea rezultatului proiectului;
- **amânarea deciziilor** astfel încât informațiile acumulate să ducă către hotărârea corectă;
- **livrarea rapidă** a produselor proiectului va avea ca efect transparența proiectului față de clientul acestuia;
- **respectarea membrilor echipei** astfel încât fiecare din acesta să simtă că activitatea desfășurată este importantă pentru proiect;
- **optimizarea întregului proiect**;

Metodologia Lean development se pliază pentru orice proiect care necesită schimbări rapide. Livrările intermediare sunt ceva obișnuit, clienții proiectului luând decizia asupra momentului livrării rezultatelor incrementale.

3.4.8 Extreme Programming

Este o metodologie AGILE care s-a dorit a fi o alternativă la metodele clasice de dezvoltare. Aceasta a fost dezvoltată de către Kent Beck. Scopul acestei metodologii este de a răspunde schimbărilor târzii din proiecte.

Ca și celelalte metodologii AGILE, Extreme Programming este o metodă incrementală, care prin livrări repetate și pași mărunți produc stabilitate proiectului.

Valorile metodologiei sunt [50]:

- simplitate;
- comunicare;
- feedback;
- respect;
- curaj;

Dimensiunea medie a echipelor de dezvoltare este de 10-12 programatori experimentați. Înainte de începerea implementării codului sursă, Extreme Programming nu pune accent pe crearea documentelor de proiectare.

Tehnicile folosite sunt:

- evaluare continuă;
- testare continuă;
- îmbunătățire continuă;

Fazele metodologiei sunt:

- explorarea;
- planificarea;
- iterații către lansare;
- producția;
- mentenanța;
- închiderea;

Cerințele sunt descrise sub forma unor cazuri de utilizare, în care sunt descrise în proză funcționalitățile sistemului. Pentru acceptarea funcționalității, clientul stabilește criteriile de acceptanță ale fiecărui caz de utilizare.

Următorul pas al modelului este de a crea un plan de lansare. Acest plan demonstrează metoda incrementală de livrare.

Particularitatea codificării este reprezentată de prezența clientului în proiect. Codul trebuie să corespundă anumitor standarde, iar prima faza a testării este reprezentată de testarea codului sursă. Testarea codului sursă este urmată de integrarea codului în sistem.

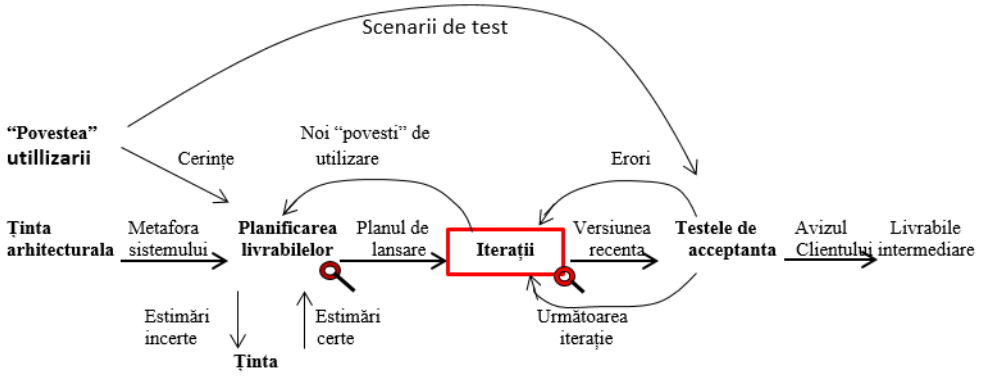

Fig. 3. 7 Procesul XP [50]

3.5 Metode de dezvoltare clasice vs. Metode de dezvoltare agile

Extinderea componentelor electronice în diferite domenii, fie ele sisteme informatice complexe sau cele simple de uz casnic, a dus la necesitatea analizei și îmbunătățirii modelelor de lucru. În proiectele software, îmbunătățirea modelelor de lucru se referă de cele mai multe ori la adaptarea și combinarea mai multor metodologii aplicate implementării unui anumit tip de proiect software. Pentru o înțelegere clară a manierei de combinare și îmbunătățire a acestor modele în continuare se prezintă o analiză comparativă a modelelor tradiționale cu cele de tip AGILE (Tabel 3.1).

Tabel 3. 1 Diferența între dezvoltare agile și cea clasica [107]

	Model tradițional	**Agile**
Ipoteza	Sistemele se pot specifica 100%, sunt previzibile, se pot dezvolta planificând fiecare activitate.	Software-ul cu standarde înalte de calitate se poate dezvolta folosind echipe de dimensiuni mici. Principiile folosite sunt cele de proiectare continuă, îmbunătățire continuă și testare. Aceste principii se bazează pe un feedback rapid la schimbare.
Control	Procesele sunt în centrul atenției	Oamenii sunt în centrul atenției
Stilul managerial	Comandă și controlează	Leadership și colaborare
Managementul cunoașterii	Explicit	Tacit
Atribuirea rolurilor	Individual, se favorizează specializarea	Echipe care se organizează intern, se încurajează schimbul de roluri
Comunicarea	Formală	Informală
Rolul clientului	Important	Critic
Ciclul proiectului	Controlat de sarcini și activități	Controlat de caracteristicile produsului
Modelul de dezvoltare	Modelul ciclului de viață (cascadă, în spirală sau variații)	Model evolutiv de dezvoltare
Structura organizațională dorită	Birocratică cu multe procese formale	Cooperantă și flexibilă
Tehnologie	Fără restricții	Favorizează tehnologiile orientate pe obiecte

De-a lungul timpului s-au evidențiat mai multe tipuri de modele. Acestea au fost categorisite astfel:

- modele tradiționale:
  - modelul în spirală;
  - modelul în cascadă;
  - modelul în V;
  - modelul în W;
  - etc.;

- modele de implementare agile:
  - SCRUM;
  - Extreme Programming;

- Lean Development;
- DSDM;
- ASD;
- Crystal;
- FDD;
- AMDD;
- etc.;

3.6 Ciclul de viață al proiectelor de tip automotive

Obiectivul paragrafului este de a prezenta ciclul de viață al unui proiect software de tip automotive. Ținând cont de faptul că proiectele software de tip automotive depind de procesele de producție, dezvoltarea acestora urmărește fazele ciclurilor de dezvoltare clasice. Aceste modele sunt modelele în cascadă, în V sau modelul în W. Deși mulți cercetători sunt de părere că modelele clasice nu au flexibilitatea necesară îndeplinirii cerințelor clienților (Yusuf și Adeleye, 2002)[115], modelele clasice oferă stabilitate și siguranță implementării cerințelor inițiale.

Chiar dacă Larmann(2004)[79] și Martin (1999)[90] considerau că metodologiile iterative se pot folosi adecvat doar în proiecte mici, în industria automotive, cu proiecte de dimensiuni mari, metodologiile iterative și-au găsit de asemenea utilizarea. În alegerea modelului de dezvoltare, coordonatorii de proiect iau în calcul următorii factori:

- modelele de dezvoltare utilizate în sistemul din care face parte proiectul;
- experiența echipelor de dezvoltare în utilizarea anumitor modele;
- adaptabilitatea modelului la cultura țării în care se desfășoară proiectul sau a componenților echipei;
- costurile alegerii unui model în defavoarea altuia;

În industria automotive se consideră că cerințele se definesc la începutul proiectului, modificarea acestora fiind puțin probabilă pe parcursul evoluției acestuia, deoarece procesele și utilajele de producție nu se modifică, în general, până la sfârșitul producției unui anumit produs.

Alegerea modelelor de dezvoltare presupune existența tuturor cerințelor înainte de a începe implementarea codului sursă.

Ciclul de viață a proiectelor se compune din înlănțuirea proceselor care vor duce la realizarea obiectivelor acestora. Fie că este vorba de procese AGILE sau procese tradiționale de dezvoltare, ciclurile de viață ale proiectelor software încep prin a defini cerințele acestuia și se continuă prin dezvoltarea proiectului până la finalizarea lui. Finalizarea unui proiect poate însemna darea în folosință sau poate include și procesul de mentenanță.

În majoritatea cazurilor, proiectele software automotive vor fi realizate de către furnizori, aplicarea rezultatului proiectului realizând-se la OEM (Original Equipment Manufacturer). Nevoia de creare a conceptului proiectului face ca startul proiectului la client să fie devansat față de startul proiectului la furnizor, deoarece concepția proiectului se realizează la OEM.

În faza de conceptualizare se vor defini într-o granularitate mare proprietățile produsului proiectului. În același timp cu începerea definirii cerințelor, proiectul este sincronizat între furnizor și client.

Primul pas în organizarea proiectului îl reprezintă analiza mediului socio-cultural în care se va desfășura proiectul. În industria automotive, internaționalizarea proiectelor este cheia care duce la crearea produselor general acceptate pe piață.

Conform Project Management Body of Knowledge [102] mediul socio-cultural este unul din factorii care au impact asupra desfășurării proiectului. Echipa trebuie să înțeleagă cum sunt afectați oamenii de către proiect și cum oamenii afectează proiectul. Această înțelegere poate consta în cunoașterea aspectelor economice, demografice, culturale, educaționale, etice, etnice, religioase și alte caracteristici care ar putea influența oamenii care sunt afectați de proiect. Managerul proiectului trebuie să examineze și cultura organizațională pentru a determina dacă rolul managerului de proiect este recunoscut în contextul cultural al proiectului.

Odată identificat mediul socio-cultural de desfășurare a proiectului cât și costurile aferente furnizorilor, se vor putea identifica resursele în funcție de complexitatea proiectului.

Este bine cunoscut faptul că la începutul proiectului și la finalul acestuia numărul resurselor este redus, în faza de dezvoltare numărul resurselor fiind maxime (Fig. 3. 8).

Un alt factor care influențează desfășurarea proiectului este impactul diferiților deținătorilor de miză „stakeholderi" la proiect de-a lungul desfășurării acestuia. Fig. 3. 9 arată că la începutul proiectului influența participanților la proiect este mare, scăzând către zero la finalul proiectului.

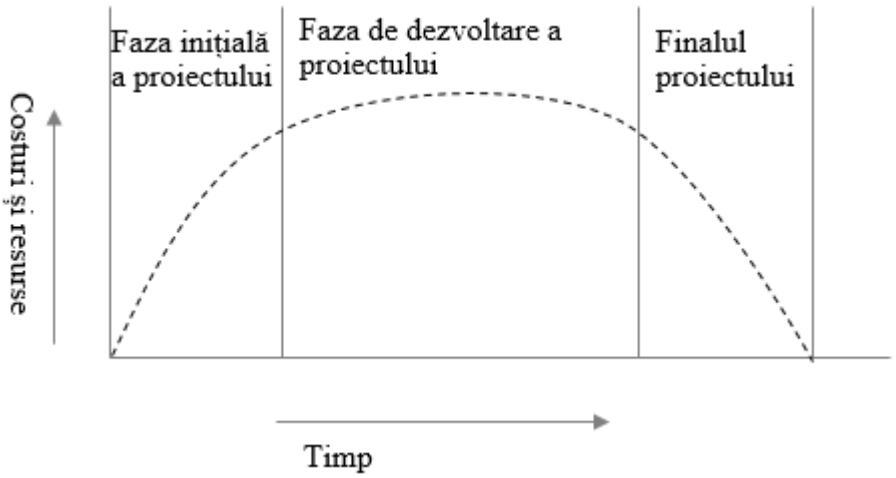

Fig. 3. 8 Nivelul costului și a resurselor umane de-a lungul ciclului de viață a unui proiect [102 – Project Management Body of Knowledge]

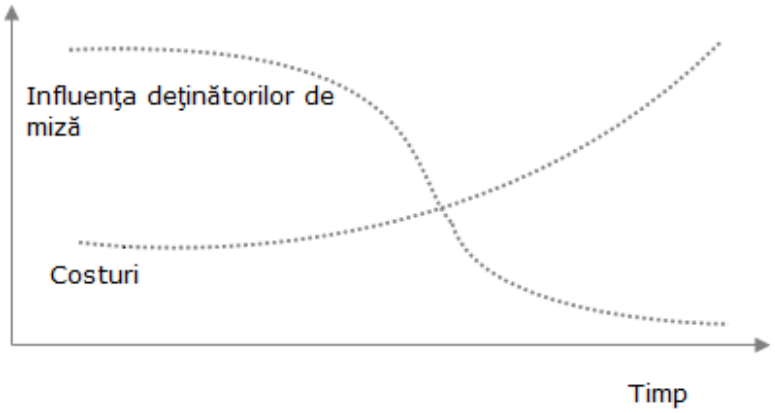

Fig. 3. 9 Influența participanților proiectului de-a lungul proiectului [102]

3.6.1 Modelul în V

Scopul paragrafului este cel al prezentării generale a fiecărei faze a modelului în V. În Fig. 3. 10 este reprezentat grafic modelul de dezvoltare în V. În limbajul coordonatorilor de proiect cu experiență, acesta se împarte în două părți:

- partea din stânga:
  - -conține fazele care descriu cerințele, arhitectura proiectului și implementarea proiectului.

- partea din dreapta modelului:
  - descrie fazele de testare ale proiectelor.

Prima etapă a părții din stânga a modelului în V o reprezintă *specificarea cerințelor*. Aflând-se în partea stângă a modelului, această treaptă include și organizarea proiectului. În etapele următoare ale laturii din stânga a modelului se specifică conținutul, arhitectura proiectului și proiectarea modulelor (Fig. 3. 11), în partea dreapta a modelului, pe fiecare nivel corespunzător, realizând-se testarea specifică fiecărei etape (Fig. 3. 12).

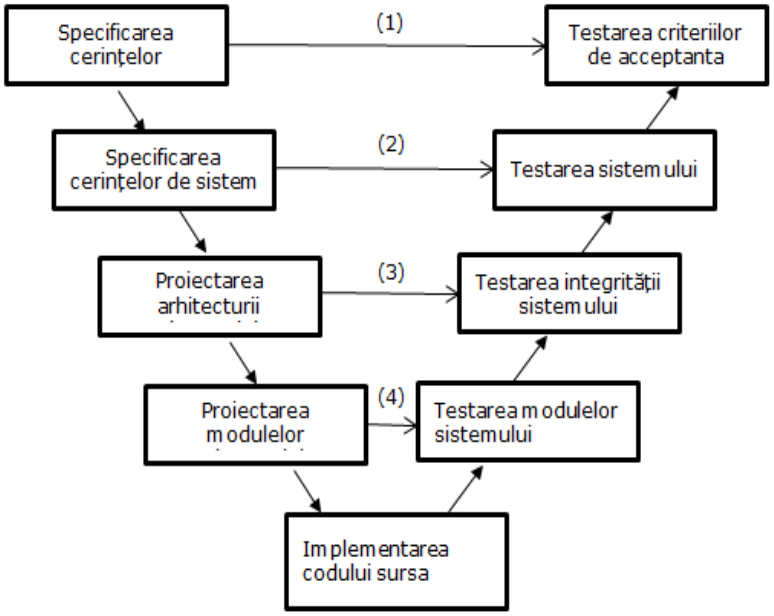

Fig. 3. 10 Modelul de dezvoltare în V. Adaptare după [54].

În concluzie, fiecărei etape de proiectare din latura din stânga îi corespunde o etapă de testare plasată în latura din dreapta(Fig. 3.10):

(1) Cazuri de testare de acceptanţă;
(2) Cazuri de testare ale sistemului;
(3) Cazuri de testare de integrare;
(4) Cazuri de testare a modulelor;

Specificarea cerinţelor: în această fază a modelului se vor defini cerinţele proiectului. Sponsorul proiectului va trimite documentaţia specificaţiilor către furnizor sub forma unui caiet de sarcini. Cerinţele vor defini mediul de dezvoltare şi rezultatul proiectului. Acest caiet de sarcini va fi recepţionat de către furnizor. În această fază toate cerinţele vor trebuie să aibă un identificator unic. Astfel se va putea urmări stadiul implementării cerinţelor.

Specificarea cerinţelor de sistem cuprinde analiza detaliată a caietului de sarcini trimis furnizorului proiectului în faza de specificare a cerinţelor. Membrii echipei de proiect ai furnizorului analizează cerinţele clientului şi creează cerinţele de sistem „interne” furnizorului. În vederea asigurării trasabilităţii proiectului, fiecare cerinţă de sistem va trebui să fie identificabilă şi să aibă corespondent către fiecare cerinţă specificată de către client.

În faza de specificare a cerinţelor de sistem, se descriu toate instrumentele software şi hardware precum şi modulele necesare realizării proiectului. Bazând-se pe rezultatele obţinute în această fază, furnizorul va realiza oferta financiară.

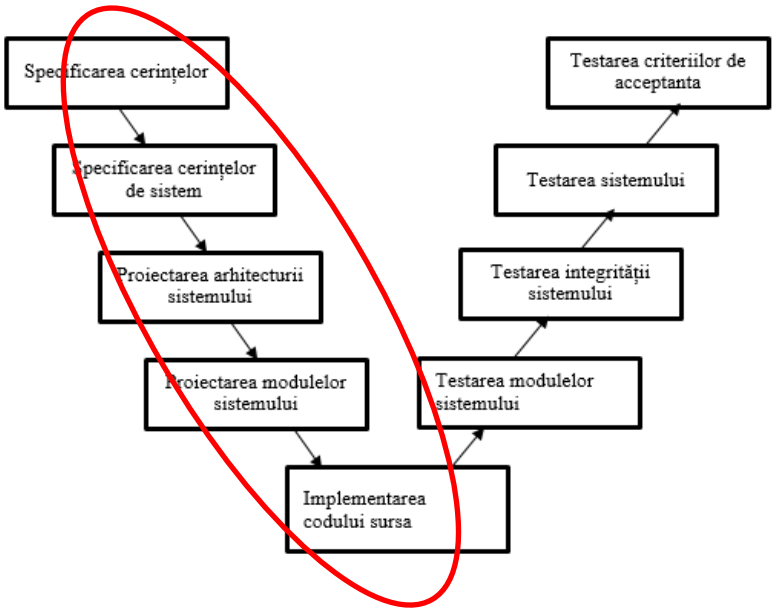

Fig. 3. 11 Definirea și implementarea sistemului. Adaptare după [54].

Proiectarea arhitecturii sistemului: se referă la definirea modulelor și interfețelor de comunicare între module. Tot în această fază se definește platforma pe baza căreia urmează să se implementeze aplicațiile proiectului, precum și o „hartă" a dependențelor între module. Aceasta va ajuta într-o fază mai târzie echipa de testare a integrității sistemului în evaluarea arhitecturii sistemului.

Proiectarea modulelor sistemului: cuprinde faza de concepere a interfețelor de comunicare din interiorul modulului. Adițional se realizează diagrama de secvență corespunzătoare fiecărui modul.

Implementarea codului sursă: în această fază se decide oportunitatea utilizării unui limbaj de programare în detrimentul altuia. Criteriile de selecție a unui limbaj de programare sunt dependente de mediul de dezvoltare, de limbajul de programare folosit pentru implementarea sistemului de operare, experiența echipei de dezvoltare cu un anumit limbaj de programare etc. Important este ca în această fază să nu se încalce cerințele arhitecturale și cele de sistem. În urma implementării codului sursă, acesta va fi supus revizuirilor.

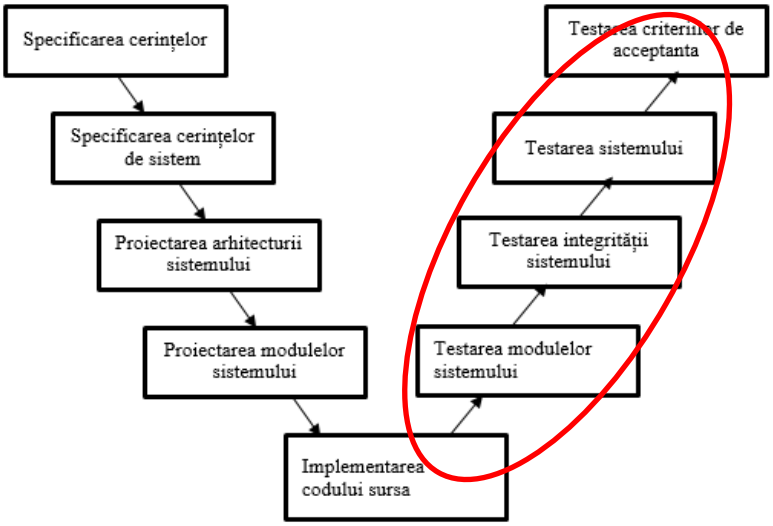

Fig. 3. 12 Validarea sistemului ciclului de viață în V. Adaptare după [54].

Testarea modulelor sistemului: este una din cele mai importante faze în testarea produsului. Aceasta deoarece permite înlăturarea de erori software în stadii timpurii ale sistemului. Cazurile de testare din această fază își au rădăcinile în faza de proiectare a modulelor. Testarea se efectuează la nivelul codului implementat [47].

Testarea integrității sistemului: are ca scop validarea arhitecturii sistemului. Testarea efectuată în această fază are ca obiectiv confirmarea funcționării comunicării între modulele sistemului.

Testarea sistemului: presupune testarea tuturor funcționalităților sistemului în urma integrării componentelor. Testarea sistemului se efectuează verificând fiecare cerință de sistem. Regula generală în testare este de a avea cel puțin un caz de testare pentru fiecare cerință. Neîndeplinirea acestei reguli va duce la insuficienta testare a sistemului. În esență testarea sistemului nu presupune testarea unei părți ale sistemului, ci a întregului sistem.[49]

Testarea criteriilor de acceptanță: este ultimul pas al lanțului verificărilor. În această fază se va testa și valida sistemul din punctul de vedere al clientului.

Fig. 3. 10 prezintă fazele modelului în V și legăturile între faze. Trecerea dintr-o fază în următoarea fază se realizează doar în urma îndeplinirii anumitor criterii de calitate. Legătura orizontală între faze se datorează necesității validării acțiunilor din partea stângă a modelului.

3.6.2 Modelul în V aplicat în proiectele automotive

În proiectele automotive, primul pas spre începerea unui proiect reprezintă realizarea studiului de fezabilitate. Scopul studiului este de a prezenta consiliului director eficacitatea proiectului și asigurarea bugetului proiectului. Specificarea cerințelor sistemului va ajuta sponsorul proiectului în evaluarea costurilor proiectului.

Asemenea descrierii teoretice a modelului în V, prima fază a proiectelor automotive o reprezintă crearea specificațiilor. Pentru a trece în faza următoare a modelului în V, rezultatul fazei actuale este utilizată drept condiție de intrare pentru următoarea fază. Aceasta deoarece una din caracteristicile principale ale modelului este reprezentată de dezvoltarea serială.

Odată generate, cerințele sistemului vor fi trimise către potențialii furnizori. Pe baza acestor cerințe, furnizorii vor trimite oferta lor de implementare a proiectului către OEM(Original Equipment Manufacturer). Analiza ofertelor se realizează pe 3 axe principale:

- axa financiară/ economică;
- axa tehnologică;
- axa timpului;

Odată cu desemnarea câștigătorului proiectului va începe schimbul detaliat de informații, în principal cel legat de cerințe, calendar și etape ale proiectului.

3.6.2.1 Faza creării cerințelor proiectului

În cadrul proiectelor de tip automotive etapa creării cerințelor proiectului (prima din latura stânga a modelului în V) include la rândul ei șase subetape, care pe baza experienței și expertizei autorului în domeniu, au fost configurate în fig 3.12.

Fig. 3. 13 prezintă într-un cadru larg procesul creării cerințelor proiectului. Cerințele sunt distribuite proiectului și subproiectelor integrate, care la rândul său sunt create datorită modularizării necesare pentru dezvoltarea proiectelor de tip automotive.

Deseori un produs automotive va fi dezvoltat pentru un concern care are în componență mai multe mărci. Proiectul va fi astfel dezvoltat încât să se conceapă mai întâi modulul comun tuturor mărcilor componente, urmând ca următoarele module să concretizeze particularitățile fiecărei mărci individuale în parte.

Procesul creării cerințelor proiectului este compus din următoarele faze principale(fig 3.13):

- **Crearea specificațiilor inițiale**: această subetapă creează practic o bibliotecă de funcționalități generice care acoperă toată paleta necesară proiectelor de tip automotive, din care se vor face în etapele ulterioare selecții personalizate.

- **Revizuirea specificațiilor**: presupune revizuirea conținutului cerințelor de către ceilalți membri ai echipei de proiect din partea clientului;

- **Crearea cerințelor finale**: în urma revizuirii specificațiilor rezultă cerințele finale ale modulului comun tuturor mărcilor, respectiv particularizările pentru fiecare marcă. Pentru o anumită marcă de

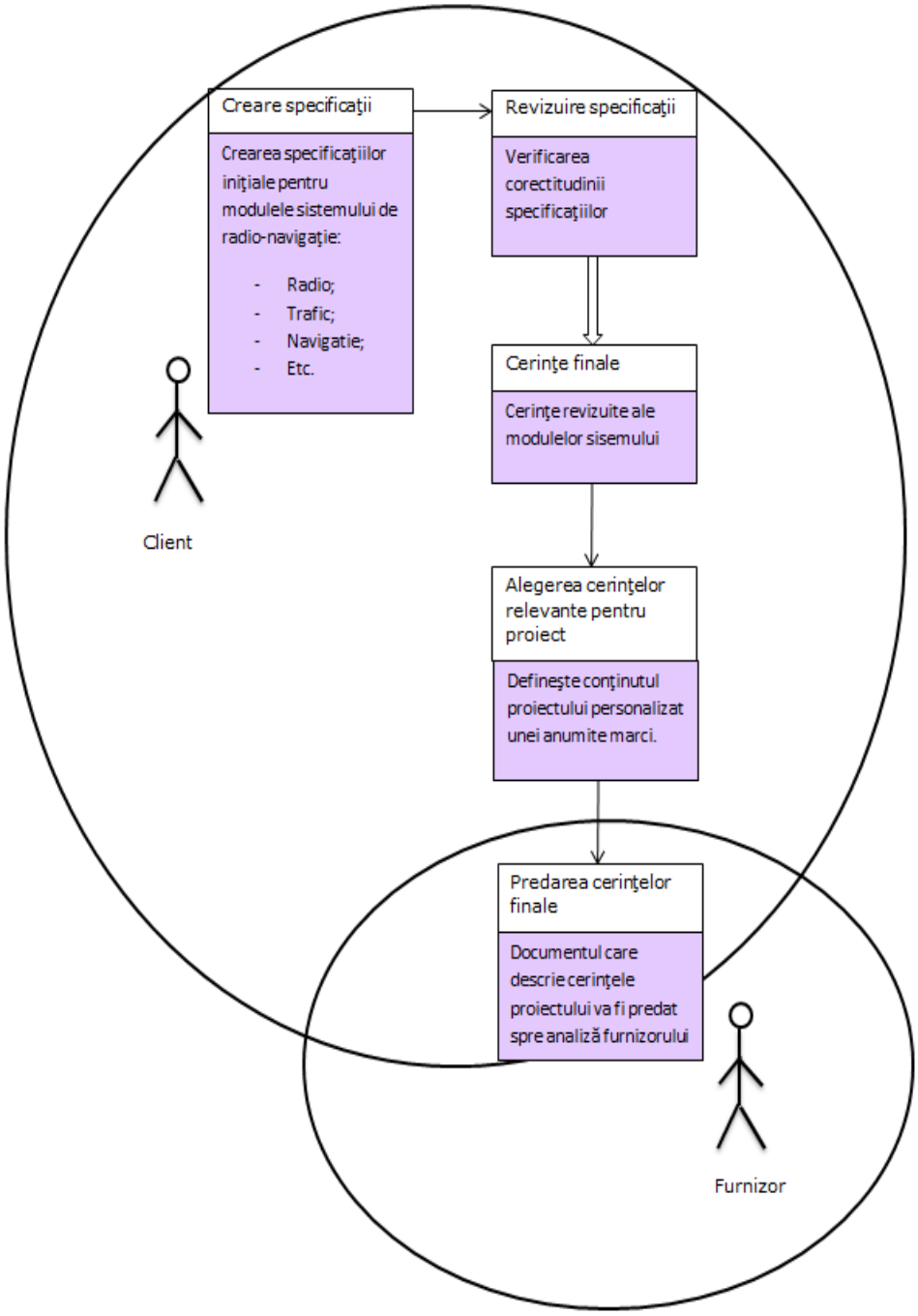

Fig. 3. 13 Procesul creării cerințelor proiectului

mașină, care este personalizată prin funcționalitățile sale se va realiza o selecție specifică din biblioteca de funcționalități generice specificată la prima etapă;

- Alegerea **cerințelor relevante** pentru proiect reprezintă faza cea mai importantă a procesului de specificare a cerințelor. În această etapă se definește conținutul proiectului personalizat unei anumite mărci;

- **Documentul** care descrie cerințele proiectului va fi predat spre analiză furnizorului. Prin predarea către furnizor a specificațiilor se încheie faza de specificare a cerințelor în cadrul modelului în V și începerea fazei de specificare a cerințelor de sistem;

Parcurgerea pașilor descriși anterior vor asigura înțelegerea și îndepărtarea cerințelor ambigue. Cerințele ambigue sunt unul din multiplele motive pentru care se vor genera schimbări multiple în timpul derulării proiectului.

Procesul specificării cerințelor, conform experienței și expertizei în managementul proiectului al autorului, durează între **6 și 12 luni**. Această perioadă se concretizează prin înlănțuirea următoarelor faze care la rândul lor au câte o perioadă de timp estimată:

- Crearea specificațiilor inițiale: **4-10 luni;**
- Revizuirea și crearea cerințelor finale: **între 3 și 6 săptămâni;**
- Alegerea cerințelor relevante pentru proiect și documentarea cerințelor finale ale proiectului: **între 4 și 8 săptămâni;**
- Predarea cerințelor către furnizor: **1 săptămână;**

3.6.2.2 Faza creării cerințelor de sistem

Specificațiile de sistem sunt o înlănțuire de procese care au ca scop înțelegerea și revizuirea cerințelor definite în faza precedentă cât și predarea cerințelor echipei de arhitecți a sistemului. Conform Hoermann et. al [76] scopul cerințelor de sistem este de a transforma cerințele clientului într-un set de cerințe tehnice care vor ghida proiectarea arhitecturii sistemului. Identificarea funcțiilor necesare arhitecturii sistemului se vor extrage din cerințele clientului și vor genera un document al specificațiilor cerințelor de sistem. În Fig. 3. 14 autorul configurează într-o formă simplificată subfazele procesului cât și activitățile care se desfășoară la furnizor sau client. Principala activitate este definită de comunicarea și transferul de informații cu scopul de a verifica fezabilitatea cerințelor. Activitățile din Fig. 3. 14 sunt descrise pe cele trei direcții posibile:

a) Furnizorul dorește schimbarea cerințelor definite în faza precedentă de către client; de multe ori cerințele clientului nu pot fi implementate tehnologic, scopul cererii modificării cerințelor este de a corecta specificațiile în așa fel încât furnizorul să poată continua cu specificarea arhitecturii sistemului;
b) Furnizorul acceptă cerințele livrate, le înregistrează în sistemul intern de coordonare a specificațiilor și informează echipa de arhitecți asupra definitivării specificațiilor;
c) Furnizorul dorește schimbarea anumitor cerințe, clientul fiind însă decidentul final dacă acceptă sau nu în funcție de importanța funcționalităților respective pentru proiectul final.

Verificarea fezabilității cerințelor, respectiv predarea spre arhitectură se realizează conform fazelor următoare (Fig. 3.14):

- **Analiza cerințelor**: este faza de analizare a cerințelor livrate de către client.

- **Revizuirea cerințelor**: rezultatul fazei de analiză a cerințelor va genera un set de specificații care să corespundă ideii furnizorului asupra arhitecturii produsului ce urmează a fi implementat.

- **Cerere modificare cerințe**: rezultă în urma analizei și revizuirii specificațiilor. În cazul în care furnizorul observă cerințe care contravin arhitecturii gândite de acesta sau observă cerințe care nu pot fi implementate din diverse motive, atunci acesta va cere clientului modificarea cerințelor.

- **Revizuire cerere modificare cerințe**: clientul va analiza cererea de modificare a cerințelor cu scopul verificării daca cererea este fondată. Factorii luați în calcul sunt[76]:
  - Cost;
  - Impact temporal;
  - Impact din punct de vedere tehnic;

- **Modificarea cerințelor**: este rezultatul acceptării cererii de modificare a cerințelor de către client.

- **Respingerea cererii de modificare a cerințelor**: această fază este parcursă numai în cazul în care clientul este de părere că cererea de modificare a cerințelor este nefondată. De cele mai multe ori în acest caz se impune reluarea procesului din faza de analiză și revizuire a cerințelor.

- **Revizuirea cerințelor modificate**: în urma acceptării de a modifica cerințele, clientul va revizui cerințele modificate astfel încât să corespundă nevoilor modulului sistemului.

- **Informare furnizor**: în această fază clientul informează furnizorul asupra deciziei de acceptare a cererii de modificare a cerințelor.

- **Livrare cerințe modificate**: presupune predarea către furnizor a cerințelor modificate.

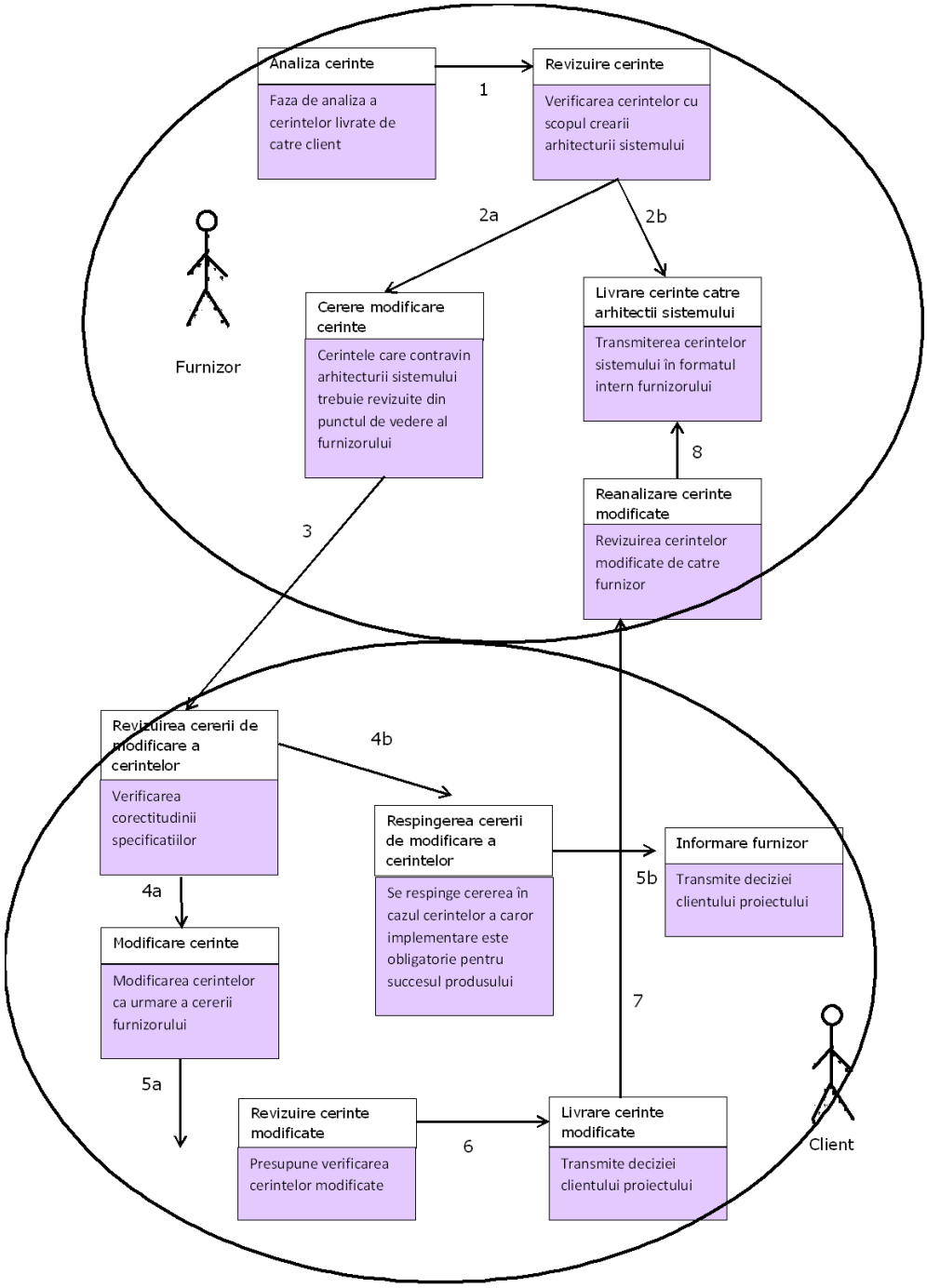

Fig. 3. 14 Procesul creării cerințelor sistemului

- **Reanalizare cerințe**: modificarea cerințelor va duce la necesitatea reanalizării cerințelor precum și a impactului asupra celorlalte cerințe deja specificate.

- **Livrare cerințe către arhitecții sistemului**: este ultima fază a procesului în care echipa de arhitecți va primi din partea managerilor de cerințe (requirement manager) specificațiile complete.

În funcție de direcția de dezvoltare a procesului de definire a cerințelor de sistem, durata acestuia poate dura **între 3 și 6 luni**.

Acceptarea formală a cerințelor din partea furnizorului autorizează definirea arhitecturii sistemului. Legătura între faza de definire și implementare a cerințelor este realizată prin intermediul unui identificator unic. Acesta permite echipei de testare să modeleze cazuri de testare pentru fiecare cerință. Totodată cerințele vor trebui făcute publice membrilor echipei proiectului cu scopul de a fi folosite drept „poartă de intrare" pentru fazele următoare. Concomitent cu finalizarea definirii cerințelor, echipele de dezvoltare trebuie să fie definite, iar pozițiile structurii proiectului trebuie să fie ocupate.
Tabelul 3.2 prezintă o modalitate de documentare a cerințelor precum și starea („status") acestora.

Tabel 3. 2 Definirea cerințelor sistemului

Identificator cerință	Titlu cerință	Descriere detaliată	Modul	Status	Acceptat?	...
Cerinta 0001	Exemplu 1	Descriere detaliata a cerintei 0001	Radio	public	Da/Nu	...
...	....	...	...	...	...	...

3.6.2.3 Faza definirii arhitecturii sistemului

În domeniul sistemelor de radio-navigație, definirea arhitecturii presupune definirea nivelelor software, a modulelor sistemului precum și a interfețelor de comunicare. Tot în această fază se vor identifica cerințele legate de arhitectura sistemului. Asemenea celorlalte cerințe, cerințele arhitecturale vor avea un identificator unic care va permite validarea acestora în faza integrării de sistem. În funcție de complexitatea sistemelor de radio-navigație, nivelurile software și modulele principale ale acestora sunt (Fig. 3. 15):

- **Hardware**: reprezintă baza arhitecturală pe care urmează a se implementa modulele software;

- **Driver**: reprezintă nivelul de bază a modulelor software care permit transmiterea informațiilor între nivelurile software superioare și hardware. Principalele module ale acestui nivel sunt:

    - grafic: permite vizualizarea interfeței om – mașină

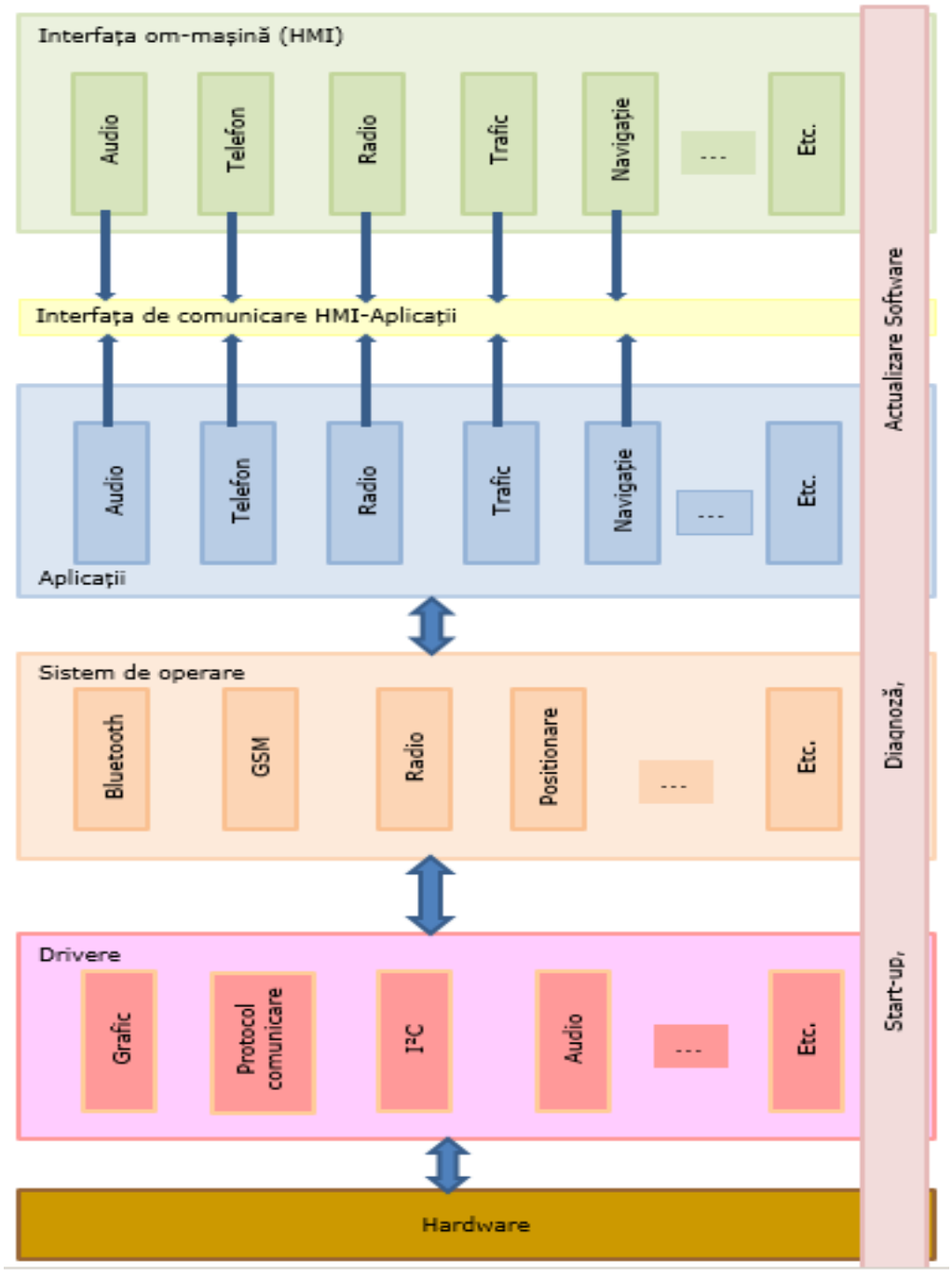

Fig. 3. 15 Arhitectura simplificată a unui proiect software automotive

  - audio: conține „codecurile” (decodificator specific formatului utilizat) necesare și permite audierea aplicațiilor bazate pe transmiterea de informații audio ;
  - magistrala de date („I2C”): este folosită în scopul comunicării între diversele componente ale sistemului;
  - etc.

- **Sistemul de operare**: reprezintă un modul care conține o colecție de programe predefinite care au rolul de „pune la dispoziție” nivelelor software superioare configurații de date elementare predefinite care asigură efectuarea operațiilor elementare tuturor sistemelor pe care le susține.
  În cadrul sistemului de radio-navigație principalele informații transmise sunt:
  - date legate de poziționarea sistemului;
  - date legate de comunicarea GSM (Global System for Global Communication;
  - date legate de comunicarea cu componente periferice exterioare sistemului (de ex. Bluetooth);
  - etc.

- Nivelul superior sistemului de operare este cel al **aplicațiilor sistemului**. Nivelul aplicațiilor are rolul de a personaliza conținutul proiectului.

- Prezentarea aplicațiilor sistemului se efectuează prin intermediul **interfeței om-mașină**. Transmiterea informațiilor privind starea aplicațiilor se realizează prin intermediul unei interfețe dedicate. Fiecare modul va avea interfață proprie de comunicare cu scopul de a asigura modularitatea sistemului.

Principalele pachete software ale nivelului software al aplicațiilor și al interfeței om-mașină sunt:

- **submodulul de navigație**: are rolul de a prezenta în mod grafic funcționalitățile navigației (harta, ruta de urmat etc.);

- **modulul de telefonie**: deservește utilizatorul cu posibilitatea de a folosi telefonul prin intermediul sistemului de radio-navigație;

- **modulul de radio**: permite utilizarea funcțiilor specifice radioului;

- **modulul de media** care redă audio diverse formate audio;

- **modulul de prelucrare a mesajelor de trafic**: presupune cunoașterea poziției, prezentarea și redarea informațiilor de trafic;

- **modulul audio**: în cazul mai multor module având funcție audio, modulul audio realizează prioritizarea activării acestora. (ex. În cazul utilizării radioului și telefonului concomitent, se redă informația provenită din modulul telefonului);

Complexitatea sistemelor este dată de numărul de nivele necesare implementării acestora, diferitele funcții având ramificații în mai multe module. Din acest motiv orice modificare adusă în modulele din nivelele „inferioare" au risc ridicat de a provoca regresii chiar în alte module. De exemplu modificări aduse în modulul de poziționare din cadrul sistemului de operare poate avea efecte atât în modulul de navigație cât și în cel al datelor de trafic din nivelul superior al aplicațiilor (Fig. 3.15). În concluzie o schimbare adusă într-un modul de nivel inferior generează modificări în mai multe module ale nivelelor superioare.

Pe lângă modulele prezentate anterior, există posibilitatea existenței unor pachete software a căror rulare au efect asupra întregii arhitecturi a sistemului. Acestea pot fi de exemplu modulul de actualizare software al sistemului sau modulul de codificare și configurare a sistemului. Existența variabilelor de configurare permite furnizorului sistemului să activeze sau să dezactiveze funcții considerate opționale.

Din punct de vedere al timpului, proiectarea arhitecturii sistemului poate dura **între trei și șase luni**, în funcție de complexitatea sistemului.

3.6.2.4 Faza proiectării modulelor

Proiectarea submodulelor unui modul al sistemului se realizează prin conceperea detaliată a interfețelor de comunicare în interiorul modulului. Documentele de proiectare a modulelor software sunt definite sub forma unor diagrame (UML) (Fig. 3.16) și descriu funcțiile care trebuie implementate. Configurarea funcțiilor se definește prin definiția clasele și metodele care vor fi implementate. Scopul procesului proiectării modulelor sistemului este de a oferi un plan pentru implementarea de software [76].

Obiectivul documentelor de proiectare a modulelor sistemului este de a transmite programatorului cerințele care trebuie implementate. De exemplu (Fig. 3.15) în cadrul submodulului de prelucrare a mesajelor de trafic, documentul de proiectare a modulelor sistemului definește:

- interfața prin care mesajele software ajung în modul;
- metodele de prelucrare a mesajelor software;
- interfața prin care mesajele software prelucrate sunt transmise către următorul modul software;

Fig. 3. 16 prezintă un exemplu de specificare a proiectării modulelor și implicațiile implementării acestora asupra arhitecturii sistemului. Delimitarea nivelurilor software este reprezentat prin intermediul liniilor întrerupte din Fig. 3. 16.

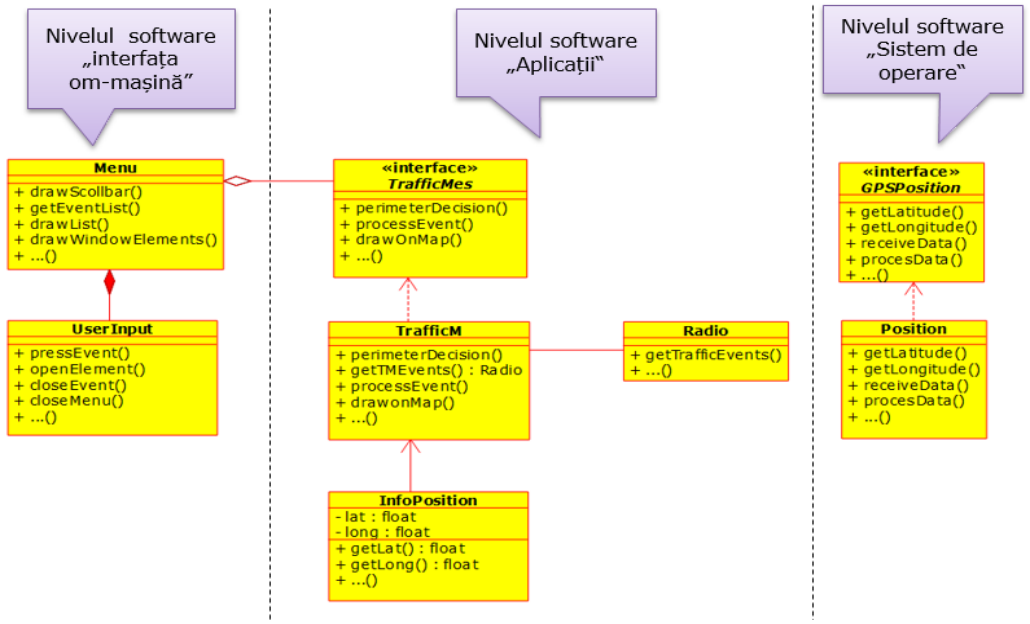

Fig. 3. 16 Specificare proiectare module software

Se observă importanța sincronizării implementării funcțiilor pe toate nivelurile de implementare. În exemplul reprezentării mesajelor de trafic în sistem, o eroare la nivelul sistemului de operare poate compromite întreaga funcționalitate. Transmiterea incorectă a poziției sistemului de radio-navigație către aplicațiile sistemului va duce la prezentarea mesajelor din altă zonă decât cea de interes a utilizatorului.

Proiectarea modulelor sistemului se realizează de mai multe echipe de dimensiuni reduse, ceea ce permite proiectarea tuturor modulelor în paralel. Timpul mediu necesar finalizării proiectării modulelor sistemului este de **4 săptămâni**.

3.6.2.5 Implementarea codului sursă

Implementarea codului sursă presupune realizarea programelor software care să respecte design-ul definit în pașii anteriori. Finalitatea fazei de proiectare a modulelor reprezintă condiția de start pentru faza de implementare a codului sursă. Implementarea software presupune respectarea unui plan de dezvoltare software. Informațiile principale care se regăsesc în planul de dezvoltare software pot fi modelate în funcție de conținutul proiectului:

- **pachetele de lucru**: definirea acestora este necesară datorită complexității proiectelor automotive. În proiectele automotive a sistemelor de radio-navigație, numărul liniilor de cod a unui modul pot depăși suma de 10.000. Astfel, împărțirea modulelor pe pachete de lucru au avantajul de a „nivela” volumul de muncă din cadrul echipelor de programatori;

- **planul temporal de implementare a cerințelor**: este bazat pe planul temporal al proiectului și are ca scop informarea datei la care se așteaptă implementarea anumitei funcționalități;
- **definirea etapelor principale ale fazei de implementare a codului sursă**: definește în principal iterațiile care vor fi parcurse în această fază;
- **obiectivele fiecărei iterații de implementare;**
- **tehnicile de implementare care trebuie urmate;**
- **informații administrative;**
- **etc**.

Rezultatul fazei de implementare a codului sursă îl reprezintă un program executabil a cărui documentare este foarte importantă. Documentarea amănunțită a codului sursă va reduce timpul necesar căutării erorilor în faza de eliminare a acestora.

Implementarea codului sursă reprezintă cea mai lungă fază a ciclului în V. Prioritatea implementării cerințelor este dată în perioada de început a fazei și nu se modifică pe parcursul delimitat de aceasta. Durata de implementare a codului sursă inclusă în faza de implementare a cerințelor, durează 1 an. Faza de eliminare a erorilor durează **6 luni**.

3.6.2.6 Testarea modulelor sistemului

Rezultatele fazei de implementare vor fi folosite ca informație de intrare pentru faza de testare a modulelor sistemului. Testarea modulelor sistemului implică crearea specificațiilor de testare precum și un plan temporal care definește fazele testării modulelor. Specificația de testare mai conține informații despre tehnici de testare manuală și automată specifică fiecărui modul al sistemului.

Scopul fazei de testare a modulelor este concretizat în vederea validării fazei de proiectare a modulelor software. Documentația fazei de testare a modulelor sistemului presupune:

- **crearea unui plan de testare** care cuprinde următoarele informații principale și poate fi modelat în funcție de conținutul proiectului:
- **identificatorul modulului într-un sistem de management al configurării**;
- **dependențe bidirecționale dintre modulul testat și alte module**;
- **probleme cunoscute**;
- **mediul de testare a modulului sistemului**;
- **instrucțiuni de utilizare a modulului**;
- **crearea de documente privind acoperirea tuturor cerințelor** de către planul de testare;

Testarea automată presupune crearea unor programe de testare a codului sursă. Decizia asupra integrării codului sursă testat în iterația următoare depinde de rezultatul testului automatizat.

Testarea manuală implică construcția și integrarea codului sursă pe baza software-ului creat în iterația precedentă a testării integrității sistemului.

Tabel 3. 3 Exemplu raport testare module sistem

Tester		Nume persoană test				
Versiunea testată		Versiune xx				
Nume modul		**Caz de testare**		**Rezultat testare versiune precedentă**		**Rezultat testare versiune actuală**
Modul 1		Caz testare 1		NOK		Parțial OK
		Caz testare 2		OK		Parțial OK
		…		…		…

Rezultatul fazei de testare a modulelor sistemului conțin următoarele informații:

- Nume persoană de test;
- Număr cazuri de test rulate;
- Număr cazuri de test rulate cu succes;
- Număr cazuri de test rulate cu erori;
- Număr de cazuri de test rulate parțial cu erori;
- Recomandarea din partea departamentului de testare asupra calității produsului;

Raportul fazei de testare a modulelor sistemului va fi trimis către toți factorii de decizie precum și coordonatorilor de proiect cu scopul planificării rezolvării problemelor raportate.

Tabelul 3.3 prezintă o modalitate de raportare a rezultatelor testării modulelor sistemului utilizată în proiectele automotive. Erorile generate în urma rulării cazurilor de testare vor fi documentate și însoțite de fișiere de trasabilitate generate de sistem în așa fel încât analiza acestor erori să se poată efectua fără a fi necesară reluarea efectuării testării de către programator.

3.6.2.7 Faza testării integrității sistemului

Faza de testare a integrității sistemului cuprinde următoarele activități:

- **verificarea documentațiilor de intrare**: are rolul de a minimiza riscul integrării sistemului, toate componentele livrate vor fi verificate. Informațiile din documentația modulelor livrate redau modificările aduse modulelor, precum și rezultatele testelor din faza de testare a modulelor.
- **integrarea tuturor modulelor într-un singur sistem**: presupune „construcția" sistemului din componentele livrate.

- **validarea integrității sistemului** se efectuează având la bază documentele de arhitectură a sistemului, având în vedere că ii corespunde acestui nivel (Fig. 3.9).
- **crearea documentației sistemului**: descrie componentele sistemului, versiunea, precum și informații referitoare la sistem ca ansamblu.

Pornind de la regula procedurilor din Automotive Spice [76], „scopul procesului de integrare al software-ului este acela de a integra unități ale software-ului în ansamble mai mari, producând software integrat, consistent cu design-ul acestuia cât și testarea interacțiunilor dintre aceste unități", activitățile detaliate din cadrul fazei de testare a integrității sistemului sunt descrise în planul integrării sistemului:

- **definirea temporală a activităților** având la bază planul temporal al proiectului;
- **strategia de integrare**: este dependentă de nivelul de dezvoltare al proiectului, aceasta putând fi o integrare în cascadă sau o integrare „all în one", în sensul că toate unitățile software-ului se integrează deodată.
    - **Cascadă**: presupune integrarea sau modificarea individuală a fiecărui modul în parte (Fig. 3. 17). Această strategie este aplicată în proiectele automotive în special la începutul proiectului.
      Dezavantajul major este dat de gestionarea complexă a componentelor sistemului precum și a efortului depus pentru a crea un sistem stabil. Strategia de integrare în cascadă pornește de la un sistem de referință în care se integrează succesiv modulele istemului rezultând astfel noul sistem de referință.

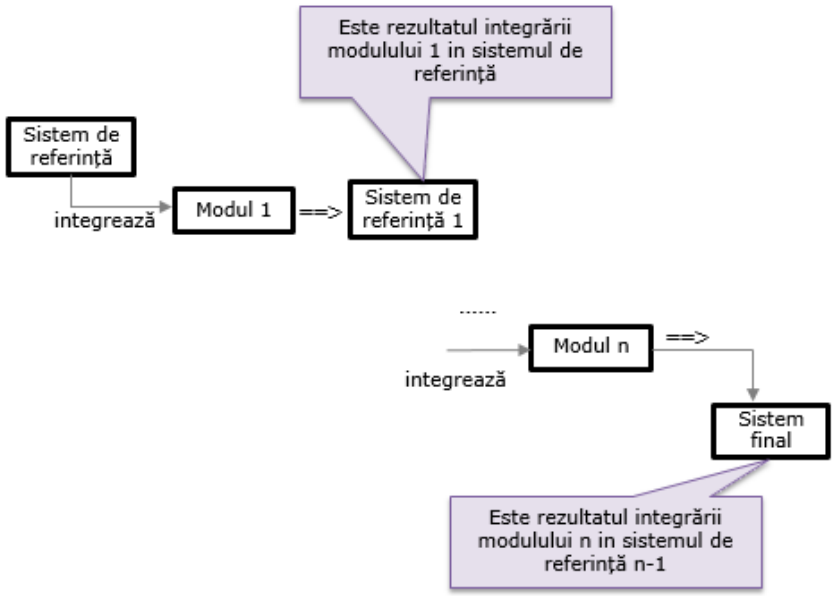

Fig. 3. 17 Integrarea de sistem bazată pe strategia în cascada [5 - Huțanu]

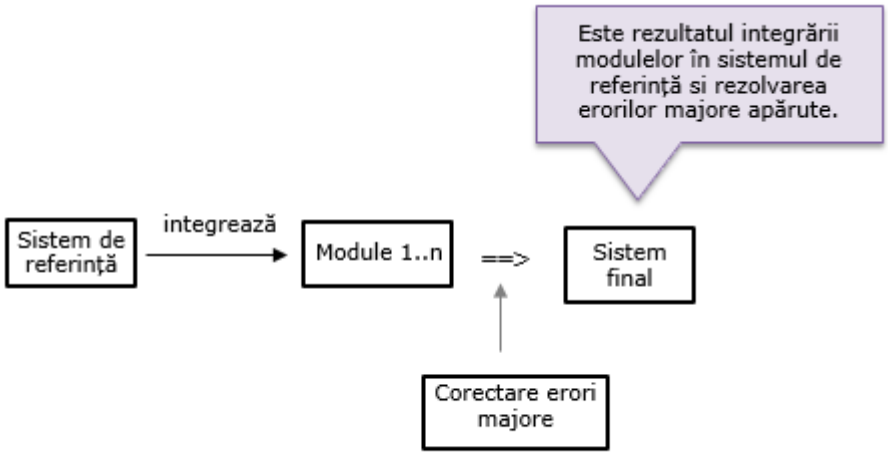

Fig. 3. 18 Integrarea de sistem bazată pe strategia „all în one" [5 -Huțanu]

- **„all în one":** presupune integrarea într-o versiune de software anterioară a tuturor modificărilor realizate (Fig. 3. 18). Această strategie implică o stabilitate crescută a sistemului. Avantajele ei sunt date de simplitatea gestionării versiunilor sistemului cât și existența disponibilității de rezolvare a erorilor majore în ciclul curent de integrare.

- **documentele de validare din faza premergătoare:** descrie documentația necesară livrării fiecărei componente.
- **definirea componentelor sistemului:** fiecare componentă a sistemului va fi caracterizată de un identificator unic (versiune) precum și descrierea componentei.
- **definirea specificațiilor de testare pentru fază de integrare a sistemului** cu scopul de a demonstra conformitatea cu arhitectura sistemului [76];
- **strategia de verificare a sistemului:**
  - -**black-box**: se referă la verificarea integrității sistemului fără a avea cunoștințe amănunțite despre interfețele din interiorul sistemului. Verificarea se realizează testând interfețele externe ale sistemului.
  - -**white box**: în acest caz se verifică inclusiv interfețele de comunicare între modulele sistemului. De obicei se folosesc diverse simulatoare cu scopul de a testa comunicarea din interiorul sistemului.

- statistica despre **gradul de acoperire a cerințelor arhitecturale** de către cazurile de test din această fază;
- definirea **strategiei de testare a regresiilor și modul de rulare a acestor teste**[76];

- **descrierea mediului de creare a sistemului**: în cazul apariției erorilor este foarte important să se cunoască mediul în care a fost creat sistemul. Aceasta va ajuta la analiza erorilor precum și la găsirea structurată a erorilor.
- **dependențe** care trebuie luate în calcul la crearea sistemului software: componentele sistemului vor fi dependente una de cealaltă. De exemplu în sistemele de radio-navigație o modificare în modulul radio poate avea efecte adverse în modulul de trafic. Din această cauză este foarte important să existe o imagine de ansamblu în ceea ce privesc dependențele între module.
- **conținutul documentației** privind sistemul nou creat:
    - identificator (versiune) unic în sistemul de gestionare a versiunilor;
    - conținutul fiecărei versiuni a sistemului software: erori rezolvate, cerințe noi implementate etc.;
    - erori cunoscute;
    - descrierea mediului de lucru;
    - diverse rapoarte și statistici;

Rezultatul fazei de integrare a sistemului va deveni criteriul de validare pentru următoarea fază din modelul de implementare. Livrabilele fazei de integrare a sistemului sunt:

- **sistemul propriu zis**;
- **documentația sistemului**;

3.6.2.8 Faza testării sistemului

Pornind de la regula procedurilor din Automotive Spice [76], „scopul procesului de testare a sistemului este acela de a certifica faptul că fiecare cerință de sistem este testată și că sistemul este pregătit pentru livrare”, definirea activităților din cadrul fazei de testare a sistemului este realizată în planul testării sistemului. Această fază de testare corespunde fazei de definire a cerințelor (Fig. 3.10). Acest document conține următoarele informații principale, pe lângă cele referitoare la rezultatul testării:

- **specificația testării sistemului**;
- **cerințele testate**: sunt acele cerințe ale sistemului care au trecut prin procesul de testare;
- **cerințele netestate**: sunt acele cerințe ale sistemului care nu au trecut prin procesul de testare;
- **strategia de testare**: definește metodele de testare. În proiectele automotive de mari dimensiuni, testele se împart în 3 categorii:
    - **testarea tuturor cerințelor**: se efectuează de obicei de două ori de-a lungul desfășurării proiectului. Prima oară când sistemul are toate funcționalitățile implementate, iar a două oară când sistemul va ajunge să aibă 0 (zero) erori;
    - **testarea selectivă** a cerințelor: se efectuează în funcție de faza proiectului. Această testare presupune testarea selectivă a cerințelor în funcție de importanța acestora;
    - **testarea de „suprafață”** presupune verificarea rapidă a principalelor funcționalități cu scopul de a găsi eventualele regresii;
- **criteriile de acceptare** sau respingere a unui test;
- **criteriile care duc la întreruperea testului**;

- **mediul de testare a sistemului;**
- **corelația între cererile de sistem și cazurile de testare;**
- **informații administrative;**

Asemenea fazei de testare a modulelor sistemului, raportarea rezultatelor fazei de testare a sistemului conține următoarele informații:

- **Numele persoanei de test;**
- **Aria testată a sistemului;**
- **Cerințele verificate;**
- **Număr de cerințe de sistem testate cu succes;**
- **Număr de cerințe de sistem reluate testate negative;**
- **Număr de cerințe de sistem testate cu rezultat parțial pozitiv;**
- **Număr de erori raportate, inclusiv numărul unic de identificare al erorii.**

Rezultatele testării de sistem vor fi transmise tuturor părților interesate, urmând ca o eventuală decizie asupra finalizării produsului să fie luată în urma verificării raportului de testare. în practică, din cauza constrângerilor de timp testarea se realizează paralel cu procesul de testare al clientului, urmând ca documentația sistemului livrat să se prezinte ulterior.

3.5.2.9 Faza testării criteriilor de acceptanță a sistemului

Testarea criteriilor de acceptare a sistemului se referă la testarea din partea clientului proiectului. Asemenea fazelor de testare anterioare, testarea criteriilor de acceptanță are rolul de a valida cerințele de sistem ale clientului.

Documentația finală presupune validarea întregului sistem integrat în sistemul clientului. În cazul proiectelor sistemelor de radio-navigație, clientul va valida funcționalitățile sistemului în mașină, semn că sistemul livrat de către furnizor este doar o componentă a unui nou sistem. Asemenea strategiei de testare a furnizorului, clientul proiectului realizează validarea sistemului pe mai multe niveluri. Aceste niveluri sunt:

- **testarea de acceptare a sistemului de radio-navigație**: această testare are ca scop verificarea funcționalității principale astfel încât componenta să nu genereze erori care duc la oprirea întregului sistem;
- **testarea completă a sistemului de radio-navigație**;
- **testarea sistemului de radio-navigație ca și componentă a unui alt sistem**;

Analiza rezultatului testării criteriilor de acceptare va duce fie la încheierea unei faze de dezvoltare, respectiv a proiectului, fie la începerea unui nou ciclu datorat erorilor existente.

Din punct de vedere temporal, fazele de verificare ale modelului în V aplicate în proiectele automotive consumă cumulat **6 luni** utilizând diferitele strategii de testare în funcție de stadiul proiectului.

Din înlănțuirea ciclurilor în V utilizate de-a lungul proiectelor automotive (Fig 3. 19) se observă diferențe între durata rulării ciclurilor, la începutul proiectului timpul necesar rulării unui ciclu fiind mai mare. Acest lucru se datorează timpului necesar organizării proiectului (aprox. 8 săptămâni), mai concret în faza de specificare a cerințelor inițiale și a implementării cerințelor se consumă mult mai mult timp, în

special pentru faptul că apare procesul de armonizare dintre client și furnizor prezentat (v. § 3.5.2.1). Pe parcursul proiectului modificările specificațiilor sau ale arhitecturii proiectului scad cantitativ și determină reducerea timpului necesar parcurgerii acestor faze (Fig. 3. 20). De altfel după terminarea fazei de construcție a funcționalităților, durata unui ciclu în V se reduce la **1 săptămână**, urmând a fi de **5 săptămâni** în cazul livrării produsului final.

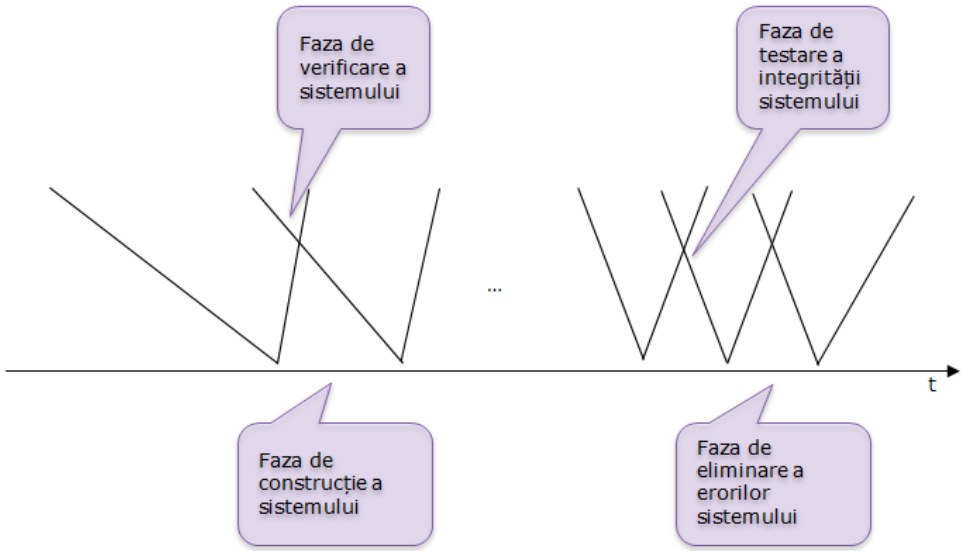

Fig. 3. 19 Reprezentare generală a duratei ciclurilor de viață pe durata proiectului

Fig. 3. 20 prezintă din punct de vedere temporal dinamica ciclurilor de viață în cazul proiectelor de radio-navigație. Este de menționat faptul că pe parcursul unui proiect automotive ciclurile de viață se reiau și se reconfigurează, suprapunându-se parțial. La începutul proiectului faza din partea stângă a ciclului în V are o durată de desfășurare mare (T1 – 8 săptămâni) urmând a se reduce pe durata proiectului (T3 – până la 3 zile). De partea cealaltă a ciclului în V, faza de verificare și validare este redusă la începutul proiectului (T2 – 2 săptămâni), urmând a crește către finalul proiectului (T4- 4 săptămâni). Din configurația ciclului în V se observă că unghiul pâlniei ciclului în V este diferit pe parcursul proiectului, unghiul la începutul proiectului fiind mai mare, urmând a scădea către finalul proiectului.

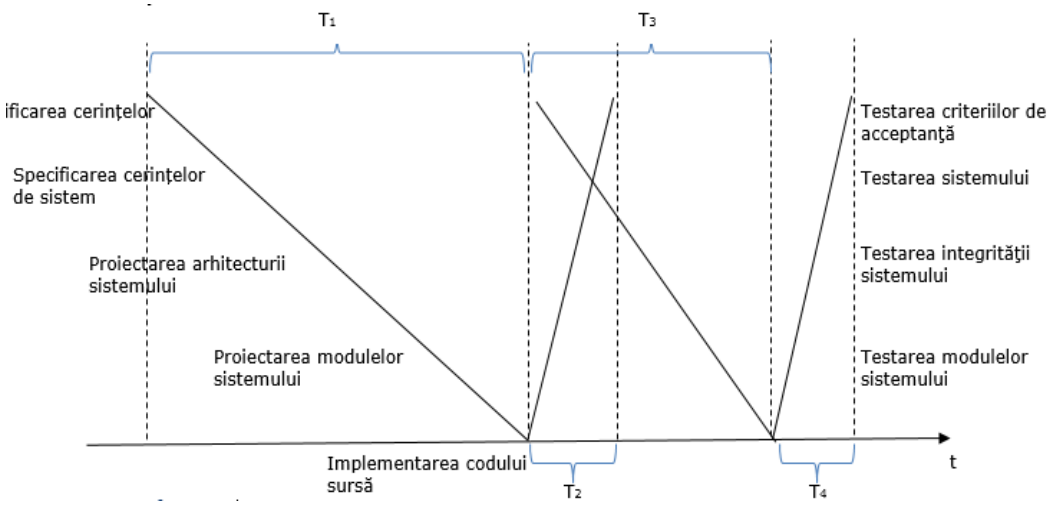

Fig. 3. 20 Dinamica ciclului de viaţă în V de-a lungul proiectului

În practica proiectelor sistemelor de radio-navigaţie, ciclurile în V consecutive se intersectează astfel încât să se economisească timp (Fig. 3. 21). Economisirea timpului se realizează prin încercarea de paralelizare respectiv suprapunere a activităţilor din 2 cicluri în V consecutive. În faza de construcţie a sistemului paralelizarea activităţilor se face pentru faza de validare şi cea de specificare a cerinţelor a 2 cicluri consecutive, urmând ca în faza de eliminare a erorilor suprapunerea ciclurilor consecutive să conţină mai multe faze ale ciclului în V, incluzând faza de proiectare a arhitecturii sistemului pe de o parte şi testarea criteriilor de acceptanţă de cealaltă parte. În Fig. 3. 21 se prezintă grafic dinamica punctului de intersecţie de-a lungul proiectului. Suprapunerea ciclurilor în V consecutive din cadrul proiectului se poate însă efectua doar cu acceptul clientului, deoarece acestuia îi va fi furnizat un sistem software netestat în totalitate, testarea sistemului realizându-se în paralel la client şi furnizor.

Livrările intermediare ale produsului software în faza de construcţie a sistemului se efectuează foarte rar, ceea ce permite în realitate rularea a 3 cicluri până la implementarea tuturor funcţionalităţilor. Durata necesară rulării celor trei cicluri de către furnizor diferă foarte mult, primul ciclu consumând foarte mult timp în faza de specificare a cerinţelor (**aproximativ 6 luni**), al doilea consumând mult timp în faza de implementare (**aproximativ 3 luni**) , iar al treilea în fazele de integrare, verificare şi validare şi rezolvare a erorilor majore (**aproximativ 1 luna**). Odată cu încheierea fazei de implementare a cerinţelor, durata ciclurilor se diminuează şi ajunge către finalul proiectului la 1 săptămână.

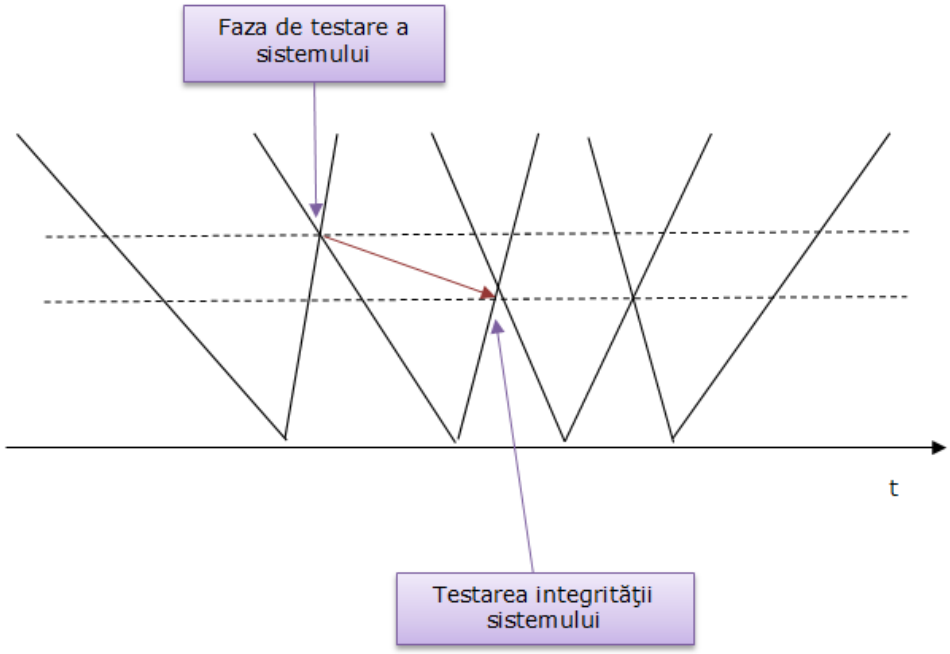

Fig. 3. 21 Intersecția ciclurilor în V în proiecte automotive

Presiunea exercitată asupra proiectului îndeamnă coordonatorii de proiect să scurteze durata de rulare a unui ciclu în V la 1 săptămână cu scopul prezentării săptămânale a progresului proiectului.

3.7 Efectele schimbărilor cerințelor asupra calendarului proiectului

Efectele modificării cerințelor existente sau crearea de cerințe noi pe parcursul desfășurării proiectului au impact diferit în funcție de stadiul proiectului. În proiectele automotive proiectul este împărțit în două faze, prima fază este cea de implementare a cerințelor iar ce-a de-a două este faza de corectare a erorilor din faza de implementare (Fig. 3. 22).

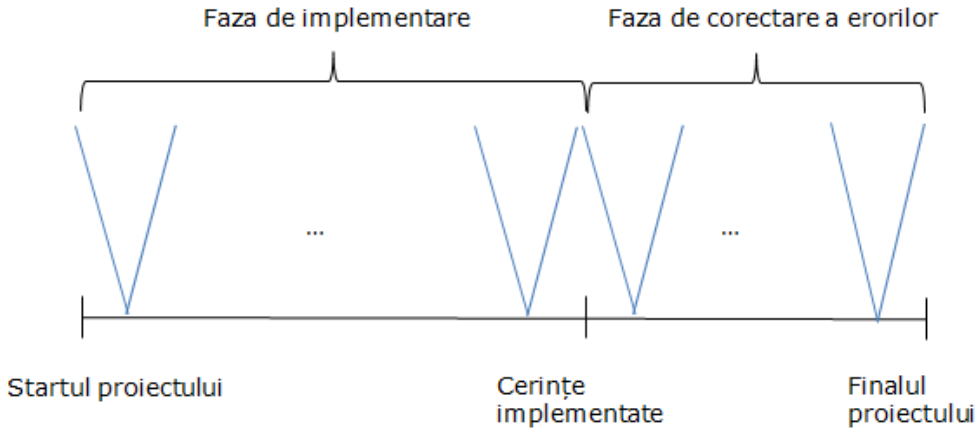

Fig. 3. 22 Structura implementării cerințelor proiectului în funcție de faza proiectului

În cazul modificărilor aduse specificațiilor se impun următoarele analize:

- Analiza efectelor asupra modelelor teoretice de dezvoltare;
- Analiza efectelor asupra modelelor automotive în stadiul implementării cerințelor.
- Analiza cerințelor asupra modelelor automotive în stadiul rezolvării erorilor de implementare

3.7.1 Efectul modificării specificațiilor asupra calendarului proiectului din teoria utilizării ciclurilor de viață în V

Motivul modificărilor aduse cerințelor proiectului își are cauza principală în tehnologiile inovatoare folosite sau create pe parcursul dezvoltării produsului. De foarte multe ori în timpul dezvoltării proiectelor pe o perioadă lungă de timp tehnologiile se vor modifica, ceea ce creează nevoia de modificare a cerințelor inițiale a proiectului. În practica dezvoltării sistemelor software automotive, efectul modificării cerințelor depinde în foarte mare măsură de stadiul proiectului. În contrast cu aceasta, în teoria dezvoltării proiectelor bazate pe ciclul de viață în V nu se ține cont de stadiul proiectului. În cazul proiectelor dezvoltate pe baza ciclului în V se consideră că orice modificare adusă cerințelor va impune reluarea ciclului curent.

În Fig. 3. 23 se prezintă înlănțuirea teoretică a ciclurilor în V de-a lungul proiectului cât și întârzierea provocată de modificarea cerințelor de două ori. În Fig. 3. 23 se simulează teoretic un proiect a cărui structură este compus din 3 cicluri în V. Cu „Întârziere A” s-a marcat influența modificării cerințelor în primul ciclu al proiectului. „Întârziere B” reprezintă întârzierea provocată proiectului în cazul modificării cerințelor în timpul parcurgerii celui de al doilea ciclu. În urma celor două modificări rezultă întârzierea cumulată a proiectului în urma modificărilor aduse cerințelor. Se observă că întârzierea cumulată este compusă din suma tuturor întârzierilor provocate în proiect.

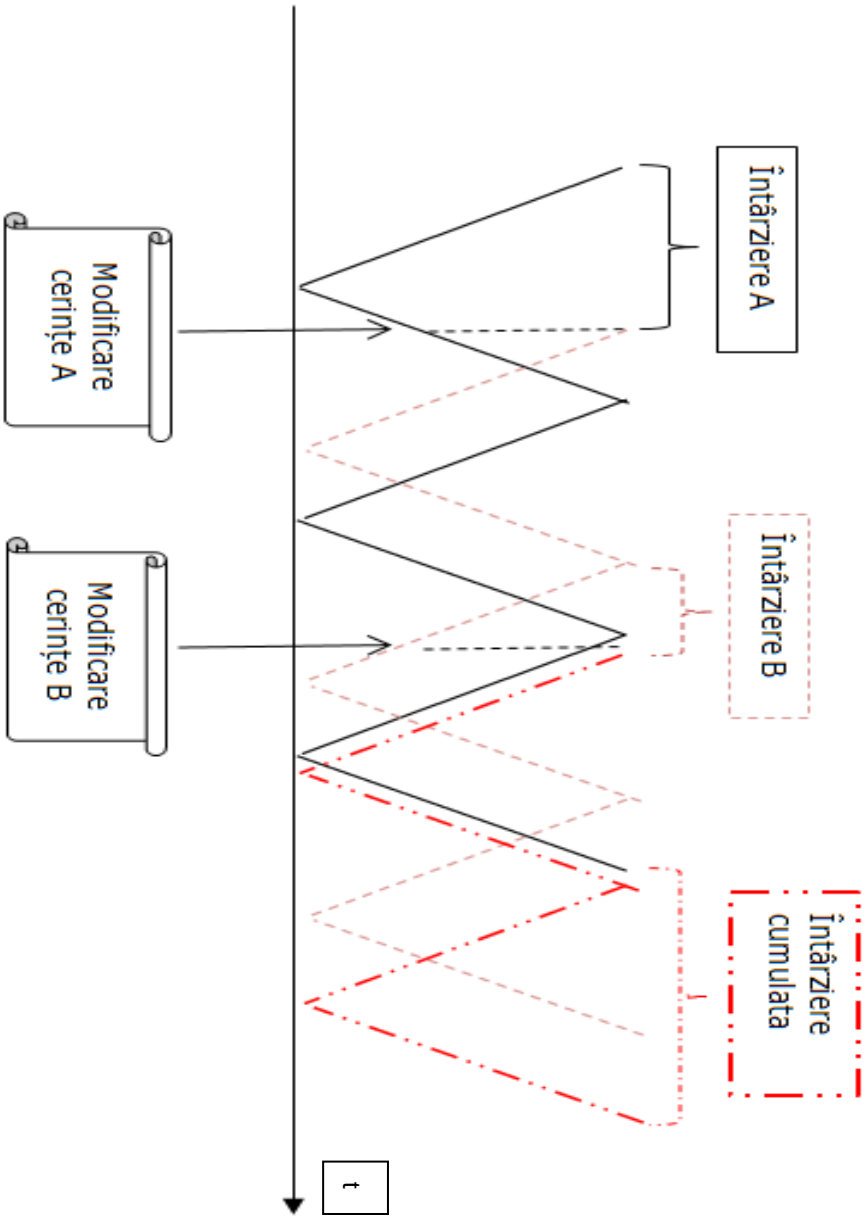

Fig. 3. 23 Traseul implementării cerinței modificate în teoria proiectelor care utilizează modelul ciclului de viață în V

Considerând „modificarea B" ca fiind o modificare care afectează mai multe niveluri de dezvoltare software ca de exemplu modulul GSM din Fig. 3. 15, acesta va

afecta calendarul proiectului. Pentru a aduce o modificare cerințelor sistemului acestea vor fi reanalizate, ceea ce determină reluarea ciclului curent. Întârzierea teoretică cauzată calendarului proiectului va fi, indiferent de gravitatea modificării, compusă din suma timpilor necesari reparcurgerii ciclului curent de dezvoltare (*Întârziere A și Întârziere B* din Fig. 3. 23).

Din punct de vedere al regresiilor aduse de implementarea funcțiilor complexe, acestea vor fi observate abia în timpul rulării fazei de testare din ciclul următor implementării noii cerințe (Fig. 3.24).

Concluzionând, aplicarea modificării cerințelor asupra teoriei proiectelor care utilizează ciclul în V va conduce la întârzieri provocate de modificarea cerințelor cât și de regresiile (v. explicație regresie în §3.5.2.4) rezultate în urma modificării cerințelor. Durata întârzierii este direct proporțională cu timpul parcurs în ciclul curent cât și cu complexitatea modificării. Cu cât modificarea implică modificarea mai multor niveluri software cu atât întârzierea este mai mare.

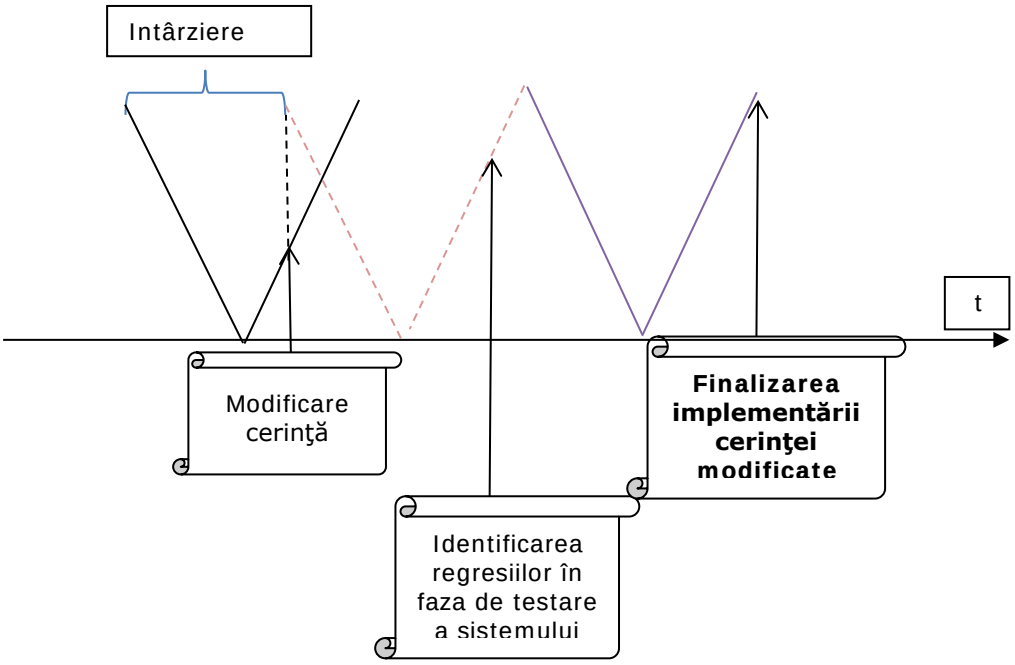

Fig. 3. 24 Traseul implementării cerinței modificate în teoria proiectelor care utilizează modelul ciclului de viață în V (adaptare după Huțanu et. al [3])

3.7.2 Efectul modificării specificațiilor asupra calendarului proiectelor de tip automotive gestionate pe baza ciclurilor de viață în V

Diferența între teoria și practica ciclurilor de viață în V constă în:

- Diferențierea stadiului proiectului (v. § 3.6).

- Metoda de abordare a modificărilor cerințelor.

În practica proiectelor automotive, modificările aduse specificațiilor vor avea impact diferit asupra calendarului proiectului în funcție de stadiul în care se află proiectul. Abordarea modificărilor va fi dependentă de faza proiectului:

- În faza de construcție a sistemului se pune accentual pe implementarea modificărilor, rezolvarea erorilor apărute în urma regresiilor realizându-se în faza de eliminare a erorilor sistemului (Fig. 3. 25).
  Fig. 3. 25 prezintă parcursul unei cerințe modificate din punct de vedere temporal. Înlănțuirea ciclurilor de viață din cadrul proiectelor automotive (v. Fig. 3. 23) fac ca atunci când se implementează o modificare de cerință, efectul acesteia să se propage pe parcursul mai multor cicluri. În cel mai dezavantajos caz, impactul modificării se va întinde de-a lungul a 4 cicluri din faza de construcție a sistemului, rezolvarea erorilor fiind planificată în faza de eliminare a erorilor sistemului.

  Șansa realizării modificării cerințelor într-un stadiu incipient al proiectului are avantajul de a contracara anumite regresii care pot apărea datorită complexității sistemului. Pașii parcurși între crearea cererii de modificare în faza de testare a modulelor și implementarea acesteia sunt:
  - **Crearea și revizuirea specificațiilor** de către clientul proiectului;
  - **Predarea noilor cerințe** către furnizorul proiectului;
  - **Analiza și revizuirea cerințelor** de către furnizor;
  - **Cererea modificării anumitor aspecte a noilor cerințe** astfel încât acestea să poată fi implementate conform structurii sistemului existent;
  - **Modificarea și revizuirea cerințelor** astfel încât să corespundă structurii sistemului;
  - **Livrarea cerințelor modificate** către furnizorul proiectului;
  - **Reanalizarea cerințelor și pregătirea acestora cu scopul analizării acestora de către arhitecții sistemului**;
  - Așteptarea rulării ciclului curent pentru eliberarea resurselor (aproximativ 1 ciclu);
  - **Analizarea, coordonarea și implementarea modificărilor** aduse arhitecturii și modulelor sistemului;
  - **Analizarea și implementarea modulelor afectate de modificare**;
  - **Implementarea codului sursă**;
  - **Testarea manuală a modulelor sistemului și documentarea rezultatelor**;
  - **Implementarea cazurilor de test** pentru testarea automată;
  - **Testarea automată** a noilor cerințe și documentarea rezultatelor;
  - **Testarea integrității sistemului**;
  - **Livrarea sistemului către client**;
  - **Testarea sistemului de către furnizor și de către client**;

- Modificările specificațiilor aduse în faza de eliminare a erorilor sistemului au un impact major asupra calendarului proiectului. Aspectul temporal în alegerea comunicării modificării este foarte important, orice modificare complexă aduce după sine noi erori, numite regresii (Fig. 3. 26). Totodată dinamica ciclului de viață în V în proiectele automotive și complexitatea modificării determină timpul necesar implementării cerințelor noi sau modificate. Dacă în faza de construcție a sistemului durata ciclurilor în V poate fi până la **6 luni**, în faza de eliminare a erorilor durata unui ciclu este redusă la **1 săptămână**. În cazul modificărilor simple, a căror durată de implementare durează un ciclu, timpul necesar implementării specificațiilor modificate este de patru cicluri respectiv **4 săptămâni**. Parcursul modificărilor este:
    - Ciclul 1: comunicarea modificării;
    - Ciclul 2: proiectarea, implementarea și testarea sistemului;
    - Ciclul 3: rezolvarea erorilor majore;
    - Ciclul 4: eliminarea tuturor regresiilor;

  Fazele parcurse până la implementarea noilor cerințe sunt aceleași ca în cazul modificărilor aduse în faza de construcție a sistemului. Diferența între cele două faze constă în viteza mărită de execuție a ciclurilor în V în faza de eliminare a erorilor, după cum urmează:
    - **Crearea și revizuirea specificațiilor** de către clientul proiectului;
    - **Predarea noilor cerințe** către furnizorul proiectului;
    - **Analiza și revizuirea cerințelor** de către furnizor;
    - **Cererea modificării anumitor aspecte a noilor cerințe** astfel încât acestea să poată fi implementate conform structurii sistemului existent;
    - **Modificarea și revizuirea cerințelor** astfel încât să corespundă structurii sistemului;
    - **Livrarea cerințelor modificate** către furnizorul proiectului;
    - **Reanalizarea cerințelor** și pregătirea acestora cu scopul analizării acestora de către arhitecții sistemului;
    - Așteptarea rulării ciclului curent pentru eliberarea resurselor (aproximativ 1 ciclu);
    - **Analizarea, coordonarea și implementarea modificărilor** aduse arhitecturii și modulelor sistemului;
    - **Analizarea și implementarea modulelor afectate de modificare**;
    - **Implementarea codului sursă**;
    - **Testarea manuală a modulelor sistemului și documentarea rezultatelor**;
    - **Implementarea cazurilor de test** pentru testarea automată;
    - **Testarea automată a noilor cerințe și documentarea rezultatelor;**
    - **Testarea integrității sistemului**;
    - **Livrarea sistemului către client;**
    - **Testarea sistemului de către furnizor și de către client**;

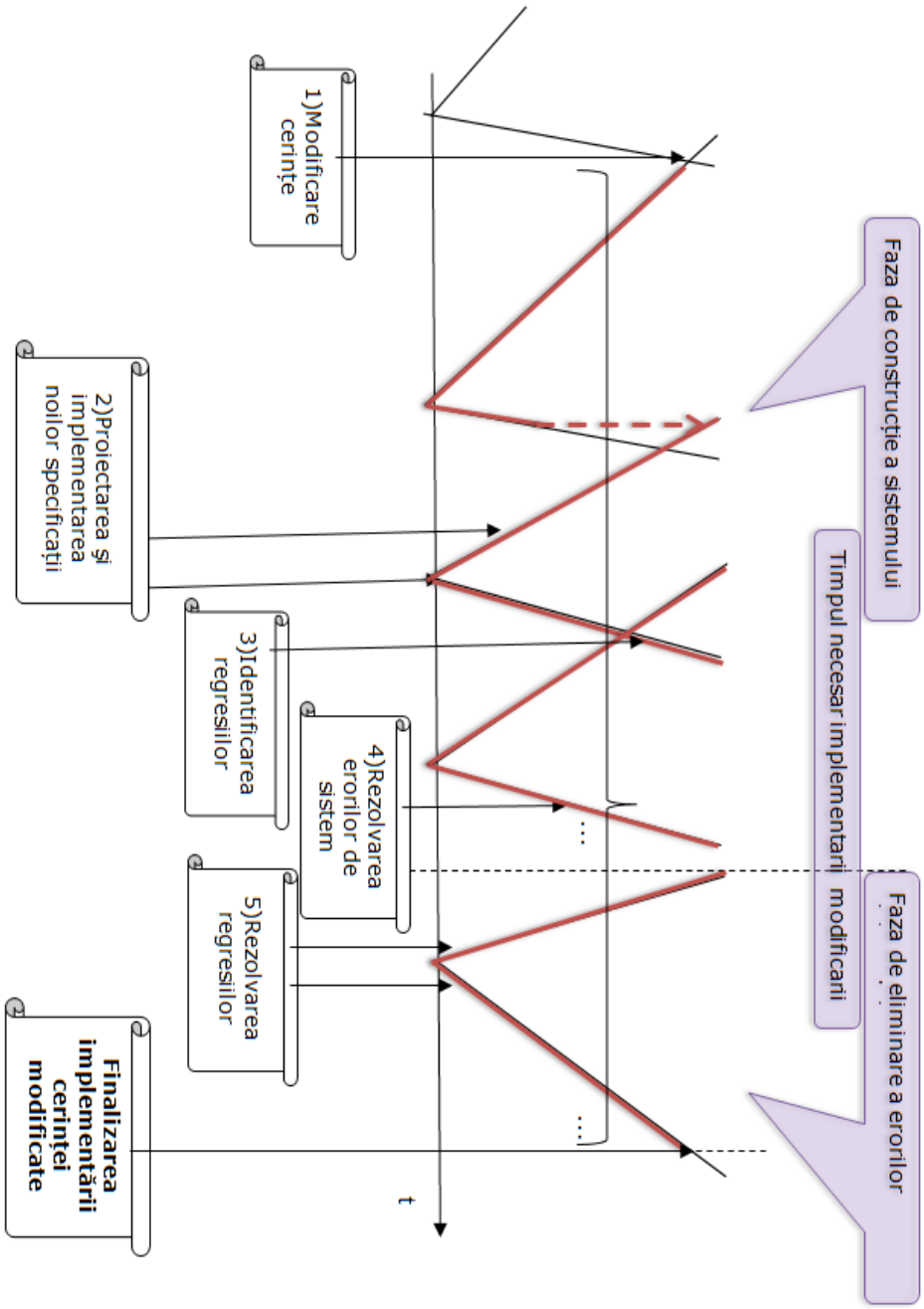

Fig. 3. 25 Parcursul cerinței modificate în faza de implementare a cerințelor proiectului

Analiza impactului asupra calendarului proiectului se realizează implicând factorul temporal al modificărilor. Din momentul apariției cererii de modificare a cerințelor din partea clientului proiectului, se pierde un ciclu de dezvoltare din cauza

faptului că pachetele de lucru ale ciclului curent au fost deja definite. Începerea implementării cerinţelor modificate se poate realiza abia după parcurgerea ciclului curent. Implementarea specificaţiilor noi în sisteme complexe implică întotdeauna şi introducerea de regresii. Pentru o înţelegere mai clară a impactului adus de implementarea modificării cerinţelor trebuie subliniat că regresia este de fapt perturbaţia adusă calendarului proiectului prin deteriorarea funcţiilor existente. Acesta este motivul pentru care coordonatorii de proiect impun un tampon de siguranţă în urma implementării unor specificaţii noi.

Impactul implementării unei modificări de cerinţă în faza de construcţie a sistemului este unul relativ redus, ciclurile de implementare existente până la finalizarea proiectului permiţând eliminarea tuturor erorilor sau a regresiilor apărute. Timpul necesar introducerii noii cerinţe în sistem în faza de construcţie a sistemului este de patru cicluri şi se compune astfel:

- Primul ciclu durează aproximativ 6 luni(v. §3.5.2.9). În această perioadă se implementează toate cerinţele sistemului şi se identifică erorile apărute în sistem.
- Al doilea ciclu de aproximativ 3 luni, necesar implementării unei cerinţe noi, este utilizat primordial în a rezolva erorile majore apărute în urma implementării cerinţei noi şi identificarea altor regresii.
- Al treilea şi/sau al patrulea ciclu este folosit în a rezolva toate regresiile şi erorile rămase în urma implementării noilor cerinţe. În funcţie de faza proiectului, durata ciclului poate dura 1 lună sau poate să fie redusă la 1 săptămână pe ciclu.

Se identifică timpul necesar implementării unei cerinţe noi comunicată în faza de construcţie a sistemului ca fiind suma timpilor necesari parcurgerii celor patru cicluri, **4,5** luni.

Impactul deciziei implementării unei cerinţe noi sau modificate asupra proiectului în faza de eliminare a erorilor este major datorită timpului redus până la termenul de predare a proiectului. Se presupune că în această fază a proiectului modificările aduse specificaţiilor nu sunt majore, astfel încât implementarea acestora să poată fi realizată în maxim două cicluri. Structura ciclurilor de implementare este identică cu cea din faza de construcţie a sistemului, durata de implementare fiind compactată, riscul crescând datorită timpului redus până la predarea proiectului şi datorită vitezei necesare de implementare a noilor cerinţe. Astfel în cel mai bun caz, durata de implementare a unei cerinţe cu o complexitate redusă este de 4 cicluri în V, respectiv 4 săptămâni(Fig. 3. 26).

Fig. 3. 27 şi Fig. 3. 28 reprezintă întârzierea provocată în proiect de modificarea cerinţelor sau din cauza rezolvării unei erori cu impact major asupra arhitecturii sistemului.

În urma modificării unei cerinţe în faza de construcţie a sistemului, întârzierea provocată calendarului proiectului este amplificată de faza de testare şi rezolvare a erorilor şi regresiilor apărute. Diferenţa între calendarul iniţial şi cel în urma modificării, precum şi etapele de întârziere provocate sunt reprezentate în paşii 2), 3), 4) şi 5) până la finalizarea implementării cerinţelor modificate din Fig. 3. 27.

Deoarece perioada de execuţie a ciclului în V în faza de construcţie a sistemului este mare, rezultatul implementării cerinţelor modificate va putea fi

observat abia în faza de eliminare a erorilor. Timpul scurs până când cerința modificată este vizibilă în sistem este prezentat în Fig. 3. 27 și Fig. 3. 28. În funcție de stadiul proiectului timpul necesar implementării unei cerințe modificate este:

- Modificarea adusă cerințelor în stadiul de construcție a sistemului:
  Timp Implementare = ∑ (durata ciclurilor utilizate în faza de construcție a sistemului; durata ciclurilor de eliminare erori);

  În exemplul dat în Fig. 3.27, timpul scurs până la implementarea completă și fără erori este:

  Timp Implementare = (3+1 luni) + (1+ 1săptămâni) = 4,5 luni

- Modificarea adusă cerințelor în stadiul de eliminare a erorilor:
  Timp Implementare = ∑ (durata ciclurilor utilizate în faza de eliminare a erorilor);

  În exemplul dat în Fig. 3.28, timpul scurs până la implementarea completă și fără erori este:

  Timp implementare = (1+1+1+1 săptămâni) = 4 săptămâni

Chiar dacă timpul de implementare a modificărilor este diferit și pare a avea un impact major asupra calendarului proiectului în faza de construcție a sistemului, riscul major este introdus în cazul modificărilor neașteptate în faza de eliminare a erorilor. Întârzierea provocată calendarului proiectului este aproape aceeași, modificările aduse proiectului în faze avansate determină aproape imposibilitatea de reacție la modificare cu scopul amortizării efectului acesteia.

Riscul asupra factorului temporal al proiectului se amplifică proporțional cu numărul modulelor sistemului afectate de modificare. Cu cât modulul afectat de modificare este într-un nivel software sau hardware inferior din arhitectura sistemului (Fig. 3. 15) cu atât numărul modulelor afectate de modificare va crește. De exemplu modificările aduse modulului de poziționare din cadrul sistemului de navigație va aduce după sine riscul de regresii în modulele de navigație, de trafic din nivelul de module software „Aplicații" (Fig. 3.15). Considerând că în Fig. 3. 27 și Fig. 3. 28 s-a adus doar o modificare simplă proiectului care a necesitat un singur ciclu de eliminare a regresiilor și comparând cu practica proiectelor automotive, unde modificările aduse specificațiilor sunt de ordinul zecilor și care afectează mai multe module, modificările vor necesita cel puțin 3 cicluri de eliminare e regresiilor.

Astfel în practica proiectelor automotive, întârzierile provocate de zecile de modificări și activități neplanificate duc la periclitarea proiectului. Lipsa instrumentelor de amortizare a întârzierilor și inflexibilitatea modelului de dezvoltare folosit va duce la nerespectarea termenului de predare a proiectului. ***În concluzie aplicarea doar a ciclului în V, atâta timp cât apar astfel de multe modificări ale cerințelor devine inoperabilă, impunându-se în mod imperativ dezvoltarea unui ciclu de viață mai complex și adaptat unor astfel de tipuri de proiecte.***

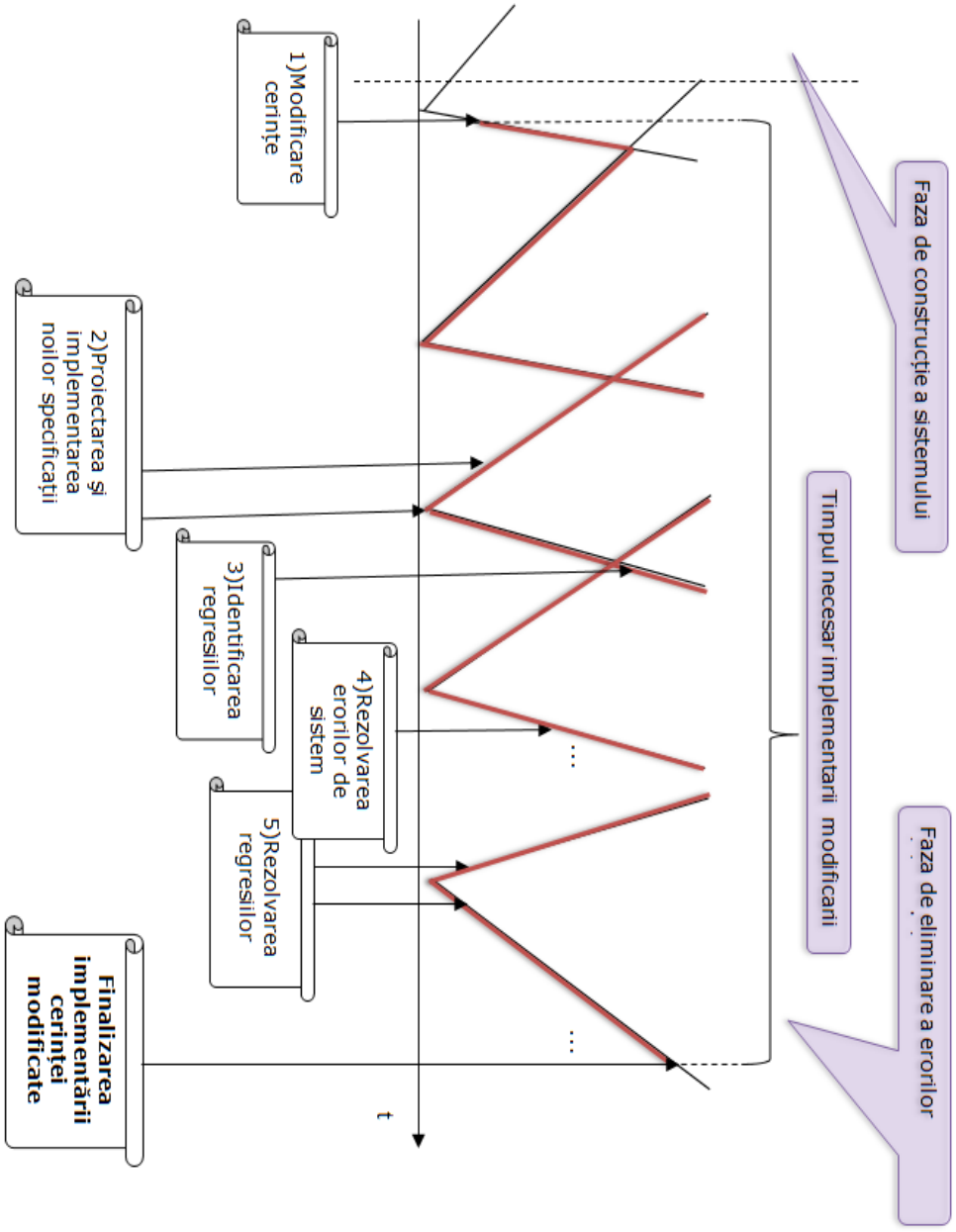

Fig. 3. 26 Parcursul cerinței modificate în faza de eliminare a erorilor proiectului

Fig. 3. 29 prezintă efectul asupra calendarului proiectului în cazul implementării a trei modificări, primele două modificări fiind compactate, adică realizate în cadrul aceluiași ciclu. În practică însă numărul ciclurilor în V din cadrul unui proiect de mari dimensiuni nu este constant, modificările multiple asupra

specificațiilor sistemului ducând la sporirea numărului de cicluri în V utilizate în proiect (Fig. 3. 30).

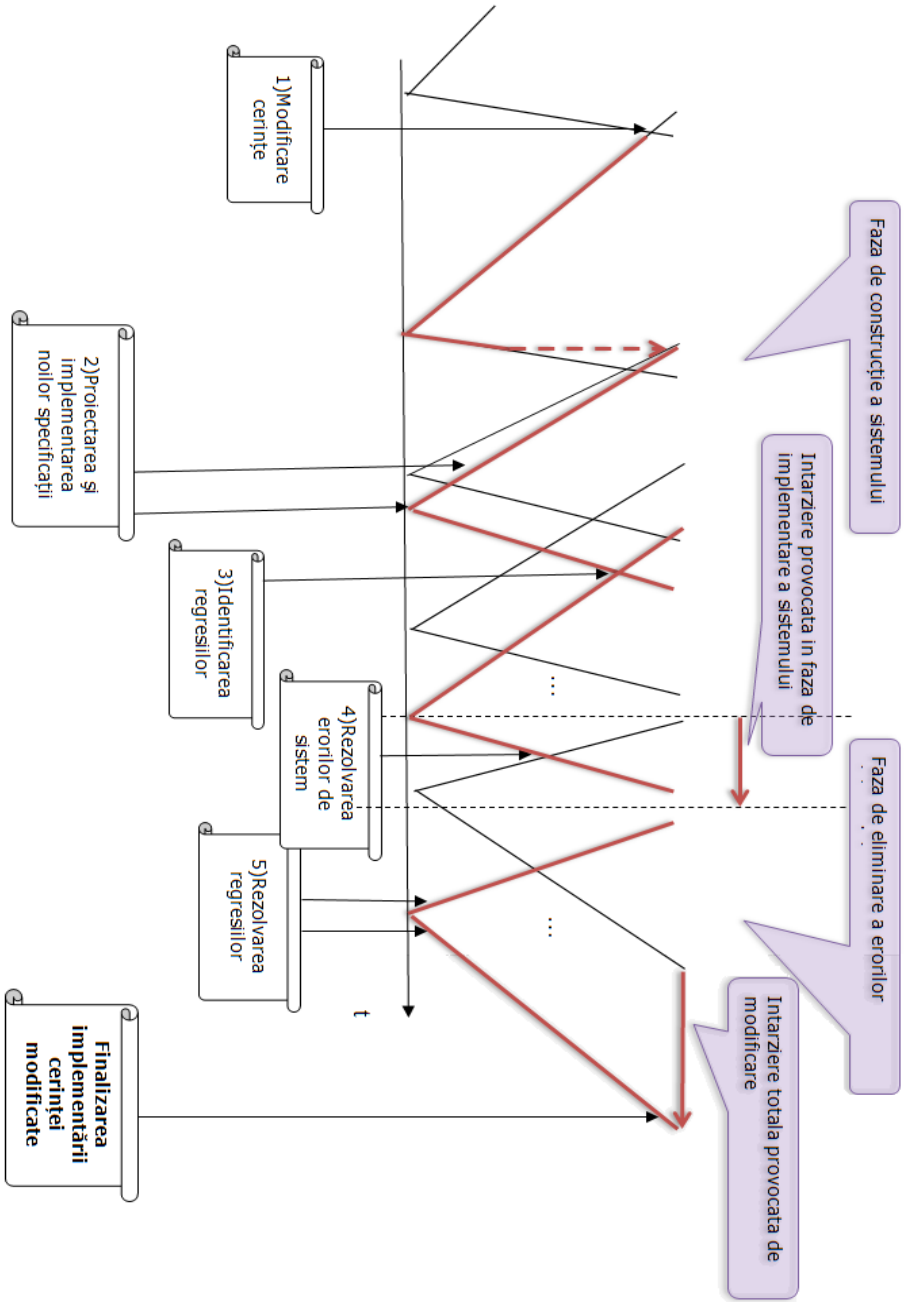

Fig. 3. 27 Efectul temporal asupra proiectului în cazul modificării unei cerințe în faza de implementare a cerințelor proiectului

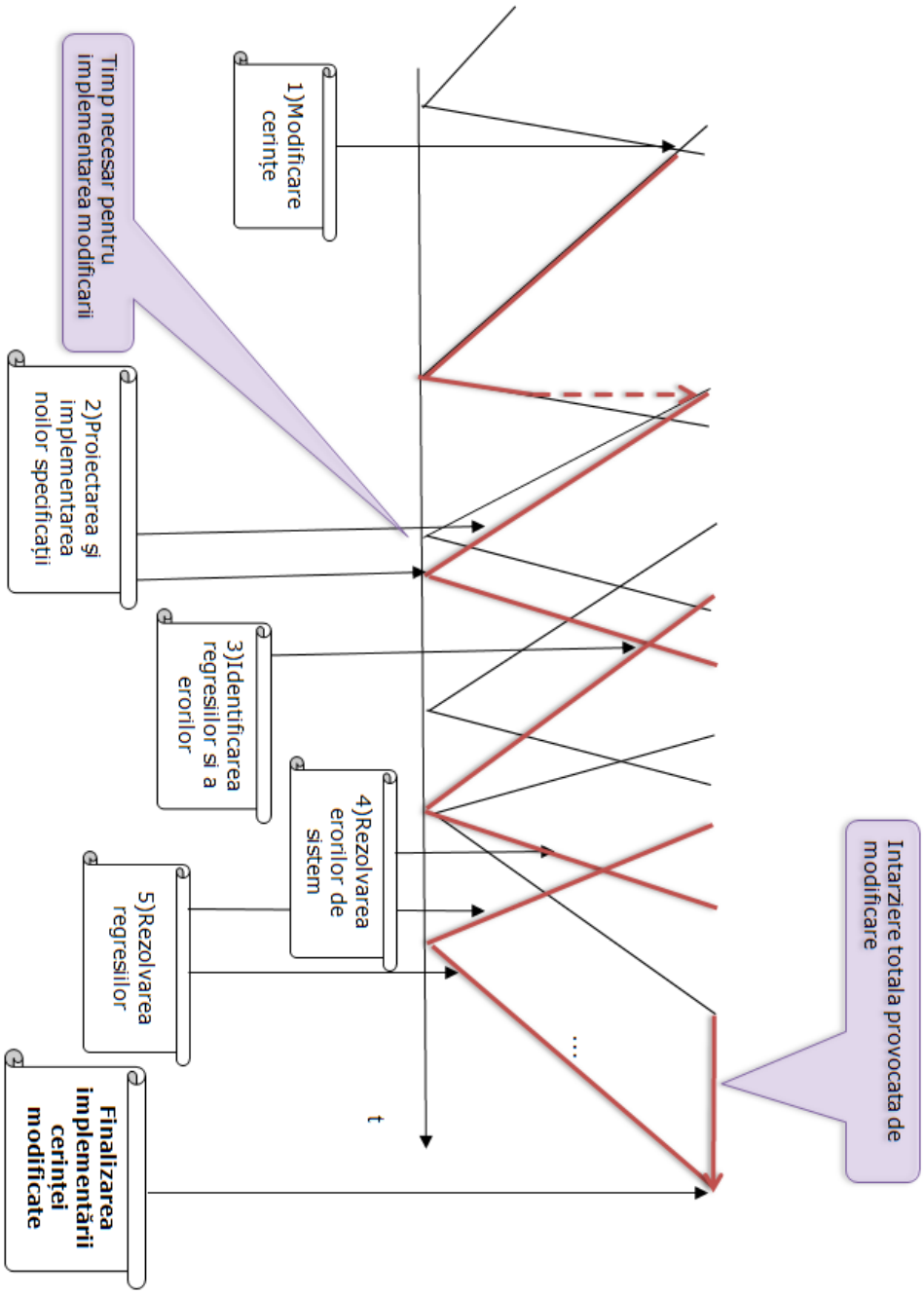

Fig. 3. 28 Efectul temporal asupra proiectului în cazul modificării unei cerințe în faza de eliminare a erorilor proiectului

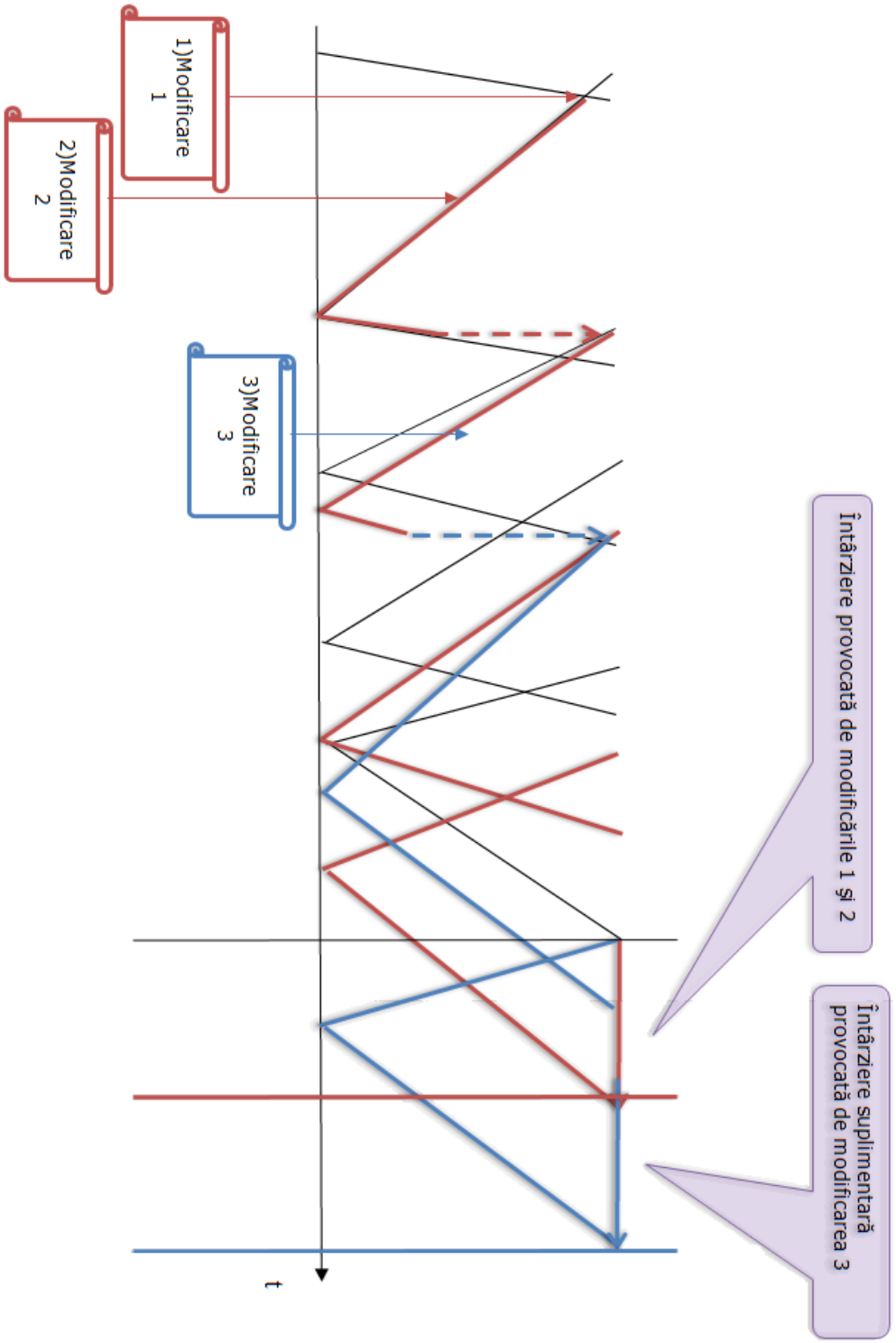

Fig. 3. 29 Întârzierea provocată calendarului proiectului în cazul modificărilor multiple

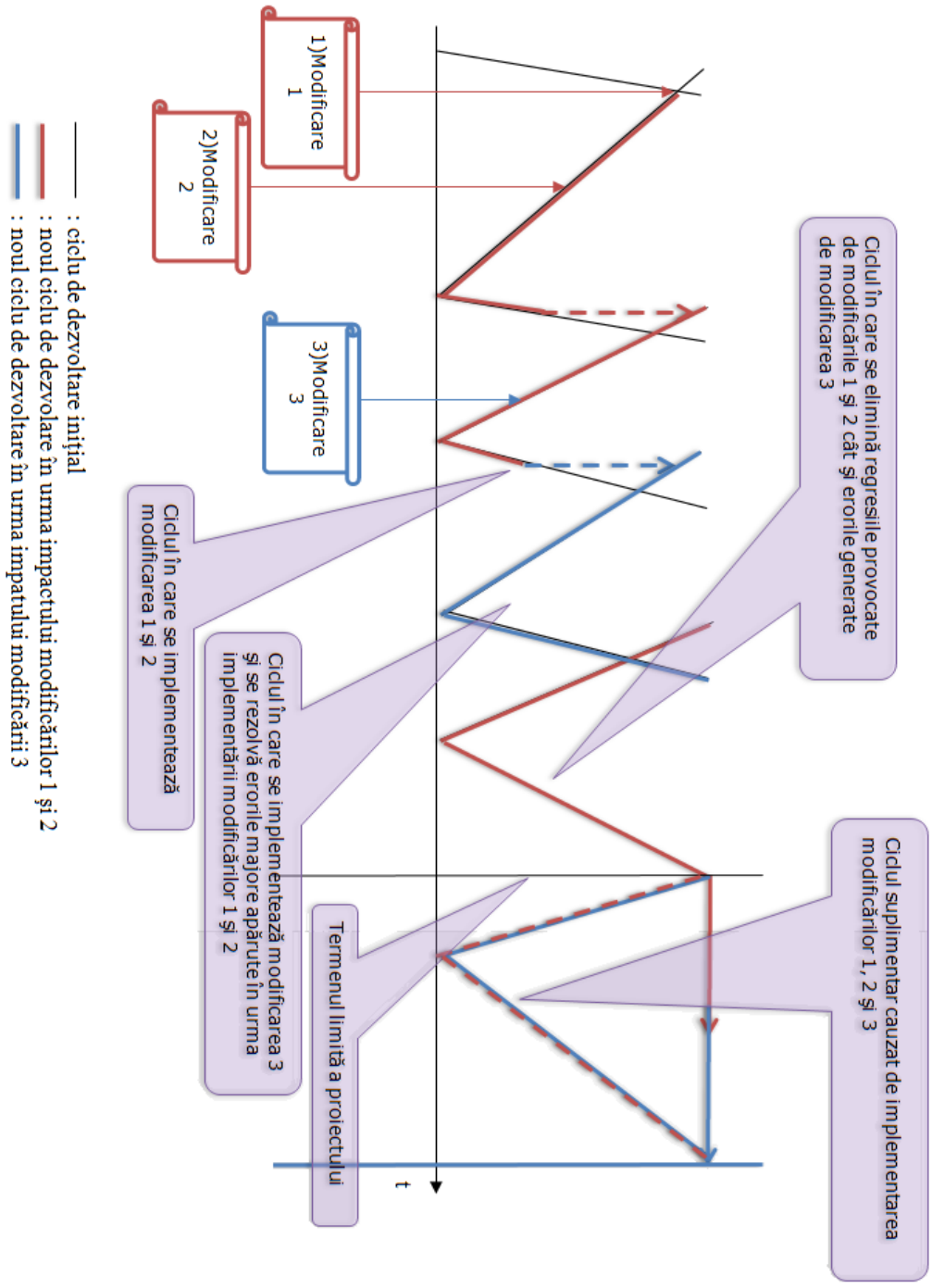

Fig. 3. 30 Reprezentarea ciclului suplimentar în urma modificării cerinţelor sistemului

Din Tabel 3. 5 și de asemenea din Fig. 3. 30 se poate identifica momentul cel mai timpuriu în care poate începe implementarea noii cerințe, adică în ciclul imediat următor. Acest lucru determină ca noile modificări să fie vizibile abia după minim trei cicluri de implementare. Faptul că finalizarea implementărilor modificărilor se realizează abia după 3 cicluri își are cauza în complexitatea sistemului și a problemelor neașteptate care pot apărea. În cazul apariției unor regresii, necesitatea rezolvării acestora va determina întârzierea implementării modificărilor cu cel puțin încă a unui ciclu. Fiecare dintre aceste modificări vor determina proiectului întârzieri suplimentare pentru care nu există soluții de compensare.

Pentru a concretiza întârzierea provocată de cele 3 modificări prezentate mai sus, sunt necesare crearea anumitor premise cât și cunoașterea desfășurării ciclului în V în faza de eliminare a erorilor din punct de vedere temporal:

- Timpul necesar rulării unui ciclu în faza de eliminare a erorilor este de **1 săptămână**;
- Timpul necesar implementării fiecărei modificări este de **3 zile**;
- Timpul necesar analizării modificărilor este de **1 zi**;
- Timpul necesar testării modificărilor este de **2 zile**;

Desfășurarea pe zile a ciclurilor în V în faza de eliminare a erorilor este prezentat în Tabel 3. 5. Fiecare ciclu este prezentat de o culoare diferită în fundalul tabelului. Diversitatea culorilor pe verticală, concretizată prin apariția culorilor diferite pe o aceeași coloană, este datorată intersectării ciclurilor consecutive, fiecare culoare aparținându-i unui ciclu implicat în intersecție.

În momentul apariției modificărilor descrise în premise, acestea consumă din planul inițial temporal al proiectului, analiza și implementarea consumând **4 zile** din ciclul în V, face ca timpul rămas pentru îndeplinirea sarcinilor inițiale să fie redus la **1 zi** (Tabel 3. 4). Chiar dacă timpul necesar testării ar fi numai de două zile, neimplementarea cerințelor inițiale va duce la necesitatea integrării unui nou ciclu la finalul proiectului, însemnând totodată neîndeplinirea factorului temporal al proiectului.

Tabel 3. 4 Reprezentarea pe zile a ciclurilor în V în faza de eliminare a erorilor

M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M
T	P/T	R/T			I	I	T	P/T	R/T			T	T														
					T	T	I	I	I			I	I	T	P/T	R/T			T	T							
														I	I	I			I	I	T	P/T	R/T			T	T
																					I	I	I			I	I

Tabel 3. 5 Reprezentarea efectului modificărilor asupra cerințelor inițiale

	J	V	S	D	L	M	M		V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M
T	P/T	R/T			I	I	T	P/T	R/T			T	T														
					T	T	I	I	I			I	I	T	P/T	R/T			T								
														I	I	I			I	T	P/T	R/T	T			T	

P=Predare intermediara a proiectului;
R=Regresii rezolvate aferente ciclului curent;
I=Implementarea corectiilor;
T=Testarea ciclului curent;

3.8 Concluzii

Ciclurile de viață reprezintă o înlănțuire de procese care conduc proiectul spre finalitatea acestuia. Adoptarea unui model de lucru în detrimentul altuia se realizează în funcție de tipul și structura proiectului, factorii esențiali în alegerea ciclului de viață fiind dimensiunea și domeniul de aplicare a proiectului.

Din analiza ciclurilor de viață tradiționale și a metodologiilor AGILE rezultă următoarele concluzii:

- cicluri de viață tradiționale, utilizate în managementul proiectelor software nu diferă în esență, acestea deosebindu-se doar în detaliu;
- cel mai des utilizat model în proiectele de tip automotive este modelul în V, acesta fiind o abordare superioară a altui model tradițional, denumit „în cascadă";
- modelele tradiționale de dezvoltare se pliază foarte bine pe proiectele în care cerințele nu se vor modifica semnificativ;
- inflexibilitatea modelelor tradiționale de dezvoltare în cadrul proiectelor de dezvoltare software au condus la dezvoltarea metodologiile AGILE;
- metodologiile AGILE reușesc să elimine slăbiciuni ale metodelor tradiționale, prin accentul pus pe comunicarea și armonizarea echipelor ori de câte ori apare o perturbație în proiect, introducând-se astfel o reală flexibilitate în abordarea schimbărilor;
- utilizarea metodelor de dezvoltare tradiționale în proiectele automotive este determinată de motive istorice, mai concret se pleacă de la faptul că producția în acest domeniu este dezvoltată conform modelului în V, proiectele preluând de la sine aceeași configurație de cicluri;
- utilizarea metodelor de dezvoltare tradiționale în producție, metode care stau și la baza dezvoltării proiectelor software automotive au la bază modele, care nu permit modificarea semnificativă a cerințelor de-a lungul dezvoltării proiectelor;

Contribuțiile personale ale autorului sunt:

- analizarea caracteristicilor principale ale metodelor tradiționale de dezvoltare a proiectelor;
- justificarea utilizării modelelor de dezvoltare în V în proiectele software din domeniul automotive;
- analiza principalelor metodologii AGILE care vin să elimine slăbiciuni ale metodelor tradiționale ;
- analiza și sinteza comparativă între procesele de dezvoltare tradiționale vs. metodologiile AGILE;
- analiza și sinteza comparativă între efectele modificării cerințelor în funcție de stadiul de dezvoltare a proiectului;
- prezentarea impactului modificărilor asupra calendarului proiectului;
- identificarea necesității dezvoltării unui nou model de dezvoltare specific proiectelor de tip automotive de mari dimensiuni.

4. CONCEPEREA UNUI MODEL DE DEZVOLTARE ORIGINAL (2JCS)

Obiectivul capitolului de față urmărește conceperea unui nou model de dezvoltare care să satisfacă cerințele proiectelor automotive de înaltă tehnologie. La baza implementării noului model de dezvoltare stau patru factori esențiali în succesul proiectelor sistemelor de radio-navigație. Primul factor este dat de decizia creării de noi tehnologii de vârf respectând calendarul proiectului. Al doilea factor îl reprezintă analiza continuă a modelelor tradiționale de dezvoltare în cazul schimbării cerințelor. Al treilea factor este reprezentat de optimizarea continuă a ciclului de viață a proiectelor software aplicând metoda teoriei constrângerilor a lui Goldratt.

Capitolul de față este structurat în trei părți, prima parte descrie cauzele schimbării cerințelor, analizează factorii decizionali care stau la baza acceptării modificărilor cerințelor precum și riscurile integrării acestora asupra proiectului.

Partea a două prezintă o metodă de calcul pentru validarea schimbării , care integrează toți factorii decizionali. De asemenea, este prezentat modul de organizare a proiectelor în faza schimbării de cerințe, fiind identificate efecte ale acelor schimbări în diferitele faze ale proiectelor bazate pe modelul în V.

În partea a treia autorul elaborează un nou model de dezvoltare a proiectelor software de radio-navigație, având la bază rezultatele analizei diferitelor modele de dezvoltare clasice precum și a metodologiilor agile.

4.1 Importanța cerințelor pe parcursul proiectelor

Schimbarea cerințelor va avea efect asupra obiectivelor generale ale proiectelor, fie că este vorba de un proiect în construcții sau de dezvoltare software. Modificarea cerințelor într-o fază târzie a proiectului reprezintă un risc major al proiectului, putând duce chiar la eșecul acestuia.

De Bakker et al. (2010)[69] susținea că în cursul unui proiect de dezvoltare software, cerințele definite inițial vor fi modificate aproape sigur, iar acest lucru va influența programul și costurile.

Din cauză că ciclul de viață al produselor de înaltă tehnologie a ajuns să fie mai mic de șase luni, principala sursă a modificării cerințelor în proiecte este dezvoltarea tehnologică. Goldratt [38] afirma că ciclul de viață a proiectelor nu mai corespunde unei curbe clasice, aceasta curbă transformându-se în profilul unei lame de fierăstrău. Apariția noilor tehnologii face ca înaintea finalizării fazei de introducere a produsului pe piață, acesta să fie considerat depășit tehnologic.

Conform normelor automotive Spice [76] principalele motive pentru care cerințele se vor modifica pe parcursul proiectului sunt:

- „Proiectul este supus unei presiuni temporale mari încă de la începutul acestuia";
- „Clientul proiectului nu poate sau nu dorește să comunice toate detaliile în fazele timpurii ale proiectului";
- „Dezvoltarea unui sistem complex inovator pentru prima dată";

- Companiile furnizoare oferă platforme pe baza cărora dezvoltă sistemele pentru diferiți clienți. De multe ori în practică aceasta generează costuri suplimentare din cauza necesității reproiectării întregului sistem;
- Numărul documentelor de specificații este deseori foarte mare (peste 100), ceea ce generează deseori contradicții între documente;

Modificarea cerințelor într-o fază avansată a proiectului va duce la sporirea riscului temporal al proiectului. Creșterea riscului cauzat de modificarea cerințelor este direct proporțional cu stadiul proiectului. Probabilitatea de acceptare a schimbărilor va fi mare în fazele timpurii ale proiectului și scade în stadiile avansate ale proiectului.

Introducerea schimbărilor într-o fază târzie a proiectului va duce la disensiuni între furnizorului și clientul sau sponsorul proiectului. „Conflictul" este generat de dorința sponsorului de a crea cele mai actuale produse. De cealaltă parte furnizorul dorește implementarea cerințelor inițiale, motivul fiind dorința de îndeplinire a obiectivul temporal al proiectului. Cu atât mai mult este necesară aplicarea unui model de colaborare în cazul lanțului de furnizori, care să fie susținut prin schimbarea de informații, sincronizarea deciziilor, alocarea resurselor comune (Badea, Proștean, Huțanu, Popa) [1] .

Un alt motiv care duce la modificarea cerințelor este definirea precară a conținutului/cerințelor încă din prima fază a proiectului. Acest lucru este recunoscut de către industrii ca fiind una din principalele cauze ale eșecului proiectelor (Chung-Suk Cho1 et al., 2001) [16].

Revenind la proiectele automotive, respectiv cele de dezvoltare a tehnologiilor noi, Milosevic D. și Patanakul P.(2005)[93] citându-l pe Eisenhardt(1989) [29], afirmau că un mediu rapid abundă de schimbări rapide și discontinue ale cerințelor, competiției și tehnologiei. În plus informația este deseori inexactă, indisponibilă și învechită.

Muhammad Nabeel Mirza et al. (2013)[94] având o atitudine conservatoare, afirmau că cerințele proiectului ar trebui să nu se schimbe de-a lungul timpului. Schimbarea cerințelor va duce la necunoașterea de către dezvoltator a ceea ce trebuie implementat, fapt ce se poate compara cu construcția unei clădiri pe o structură mișcătoare.

În teoria managementului de proiect, afirmația lui Muhammad Nabeel et. al(2013)[94] are deplină acoperire. În realitate însă definițiile specificațiilor se vor modifica de-a lungul realizării proiectelor. În domeniul automotive, pe durata execuției proiectului, apariția tehnologiilor noi vor duce la modificarea cerințelor.
Hall et al. (2008)[41] a sugerat că o planificare temporală și bugetară realistă combinată cu schimbări ale cerințelor sunt factori de risc majori pentru proiectele de dezvoltare software.

În implementarea sistemelor de radio-navigație dezvoltarea proiectelor pe baza modelului în V este o practică comună. Acesta este folosit preponderent din cauza controlului care îl oferă asupra fiecărei faze a modelului.

În contradicție cu procesele automotive, Yusuf și Adeleye (2002)[115] au studiat și apoi afirmat că firmele clasificate ca fiind agile au prezentat o performanță mai buna decât celelalte firme. Firmele neproductive se concentrează asupra măsurilor de eficientizare fără a lua în seamă cerințele clienților.

Deoarece fiecare model prezintă avantaje și dezavantaje Eppler (2003)[32] a emis ambele definiții ale calității – îndeplinirea cerințelor clienților și îndeplinirea cerințelor funcționale. Astfel a reușit să îmbine îndeplinirea așteptărilor dar și cea a cerințelor proiectului.

M. DeBellis, C. Haapala(1995) [85], C. Hood, S. Wiedemann, S. Fichtinger, U. Pautz (2008)[15] au afirmat că provocarea cea mai mare este evoluția rapida a cerințelor, care deseori tind să devină învechite. Cerințele fiecărui proiect se vor modifica de-a lungul dezvoltării acestuia, iar gradul de influență asupra programului proiectului diferă în funcție de ciclul de viață ales.

În contextul schimbării cerințelor, B. Regnell et al.(2005)[10] sublinia diferența între proiectele de dezvoltare având un singur client și cele în care piața influențează dezvoltarea cerințelor proiectului.

Deși Regnell et al. (2005)[10] încadra proiectele în două categorii, proiectele sistemelor de radio-navigație sunt o combinație între aceste două posibilități. În acest caz există un sponsor unic, însă acesta reacționează în funcție de cerințele pieței.

Clientul va schimba cerințele proiectului în momentul apariției unei tehnologii noi, astfel încât componentele dezvoltate să cuprindă cele mai noi tehnologii. Companiile dezvoltatoare de proiecte software trebuie să trateze cerințe care se dezvoltă foarte repede și care până la sfârșitul proiectului vor fi eliminate din caietul de sarcini(M. Weber, J. Weisbrod; 2003 [87]).

Competitivitatea foarte mare din domeniile tehnologiilor noi generează schimbarea cerințelor. Afirmația autorului este confirmată de H. Merisalo-Rantanen, T. Tuunanen și M. Rossi(2005) [92].

Deși modificarea cerințelor în domeniul tehnologiilor avansate este o necesitate, diferite studii au arătat că instabilitatea cerințelor este o problemă majoră a proiectelor (G. Stark, A. Skillicorn, și R. Ameele,1999[36], Y. Malaiya și J. Denton,1999[116], D. Pfahl și K. Lebsanft,2000 [24], D. Zowghi și N. Nurmuliani,2002[25]).

Furnizorul sau executantul proiectului de radio-navigație nu dorește să implementeze noile cerințe deși modificările cerințelor sunt în mare măsură dictate de către cererile pieței. Acest lucru se datorează factorului temporal al proiectelor și a importanței respectării termenelor intermediare (Huțanu et al. 2013 [5]). Densitatea mare de modificări ale cerințelor este un risc mare asupra proiectului, risc ce trebuie tratat într-un mod structurat (M. Weber, J. Weisbrid, 2003 [87]). Prima măsură a furnizorului pentru implementarea cerințelor noi este solicitarea unei liste de priorități a cerințelor.

Prioritizarea cerințelor se face în funcție de mai mulți factori(Huțanu et al, 2014 [6]):

- impactul schimbării asupra pieței;
- riscul asupra proiectului:
  - numărul liniilor de cod modificate;
  - numărul de iterații software până la sfârșitul proiectului;
- timpul necesar pentru implementarea și testarea noii cerințe;

Influența modificării cerințelor asupra dezvoltării proiectelor sistemelor de radio-navigație este un fapt dovedit de către toți cercetătorii în domeniu. Totodată cercetătorii au concluzionat că nu există proiect a cărui cerințe să nu se schimbe de-a lungul implementării.

Implementarea cerințelor pieței presupune ca între clientul și furnizorul proiectului să existe o comunicare foarte bună. Neimplementarea cerințelor cerute

de piață pot transforma un proiect într-un eșec din cauză că livrabilul proiectului nu va găsi utilizare. Din experiența autorului, soluția la astfel de probleme este o combinație între comunicare și stabilirea priorităților implementării cerințelor astfel încât produsul final să aibă utilizare.

4.2 Metamorfoza proceselor din industria automotive

Utilizarea metodelor tradiționale de dezvoltare a sistemelor software în domeniul automotive își găsește motivarea în trecutul dezvoltării componentelor automobilelor. Motivul constă în dezvoltarea târzie a sistemelor electronice în comparație cu cele mecanice. Din istoricul proiectelor automotive se observă că primele automobile construite aveau ca bază de funcționare doar componente mecanice. Sistemele mecanice, fiind 100% predictibile, iar activitățile de producție putând fi ușor planificate, au dus la folosirea modelelor tradiționale de dezvoltare.

4.2.1 Optimizarea lanțului de producție

Optimizarea proceselor domeniului automotive consta în principal în optimizarea producției. Creșterea calității și cantității producției coincidea cu îmbunătățirea procesului. Goldratt [37] a pus bazele optimizării proceselor în producție, optimizare care a fost preluată mai apoi și în proiectele de dezvoltare software. În urma analizei proceselor de producție, Goldratt a reușit să exprime în teoria constrângerilor (TOC) principalele impedimente în procesul producției. De asemenea teoria constrângerilor definește și metodele prin se pot aduce îmbunătățiri proceselor de dezvoltare.

TOC s-a dovedit a fi o tehnică folosită în managementul de proiect, care încearcă să rezolve constrângerile între factorul financiar și cel de calitate a produsului.

Primul impuls al managerilor în încercarea de a optimiza un lanț de operațiuni era de a optimiza fiecare operațiune în parte. Noutatea adusă de Goldratt[37] este privirea în ansamblu al procesului de lucru și optimizarea acestuia. Analizând un lanț de procese consecutive și interdependente, se poate observa că procesele componente au un timp de realizare diferit.

Îmbunătățirea proceselor, care au un timp de realizare redus, nu duce la optimizarea întregului lanț al producției.

În continuare este prezentat un exemplu simplu prin care se prezintă diferența între optimizarea clasică și optimizarea prin aplicarea teoriei constrângerilor (TOC) a lui Goldratt. Pentru o înțelegere mai ușoară se consideră un exemplu care nu folosește procese paralele și nu utilizează aceleași resurse pentru mai multe procese.

Se consideră un lanț de procese(P_1-P_5) de producție cu timpi diferiți de realizare. Fiecare din aceste procese este realizat de echipe unice.

Fig. 4. 1 prezintă lanțul proceselor în cazul în care timpii de execuție a proceselor ar fi egal. Optimizarea proceselor în egală măsură ar duce la îmbunătățirea de ansamblu a producției. Acest caz este însă unul pur teoretic. Din cauza existenței diferiților factori precum cel al resursei umane, nu există posibilitatea ca timpii necesari efectuării anumitor pași să fie egali. Un exemplu simplu de risc al resursei umane este dat imposibilitatea respectării termenelor contractuale pe motive medicale

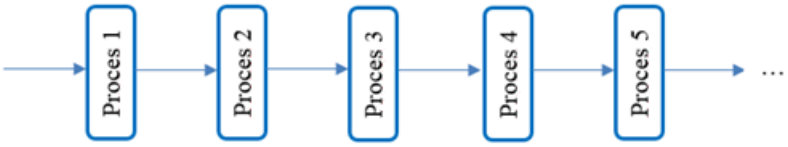

Fig. 4. 1 Lanț de producție teoretic.

Trecând la spectrul real din producție, în care timpii de execuție sunt diferiți pentru fiecare proces, se observă posibilitatea creării de stocuri pe parcursul lanțului producției. Stocul reprezintă blocarea resurselor investite, fie ele sub forma resurselor materiilor prime, fie sub forma timpului investit. Din acest motiv, crearea de stocuri în producția modernă este de evitat.

Fig. 4. 2 prezintă grafic rezultatul lanțului de producție în cazul în care procesele de producție au timpi de realizare diferiți.

Goldratt [37] a explicat metoda teoriei constrângerilor prin împărțirea procesului de optimizare în 5 pași ciclici.

- identificarea constrângerilor: este pasul în care se identifică procesul care provoacă constrângerea. Se definește constrângerea ca fiind acel pas din cadrul procesului care determină capacitatea lanțului procesului;
- exploatarea constrângerii: reprezintă exploatarea la 100% a resursei considerată cu capacitatea cea mai redusă;
- subordonarea capacității celorlalți pași din proces la nivelul constrângerii: reprezintă reducerea capacității celorlalte procese din lanțul sistemului în așa fel încât să se evite crearea de stocuri. Prin evitarea stocurilor se evită blocarea resurselor financiare.
- elevarea constrângerii: are ca scop îmbunătățirea nivelului producției. Întotdeauna capacitatea de producție a constrângerii este cea care determină capacitatea de producție a sistemului. Prin creșterea capacității constrângerii va crește capacitatea întregului sistem de producție;
- identificarea noii constrângeri a sistemului: TOC presupune reluarea ciclică a pașilor descriși anterior. Identificarea noii constrângeri a sistemului indică reluarea proceselor anterioare;

Teoria constrângerilor poate fi descrisă utilizând proprietățile mecanice a unui lanț. Constrângerea sistemului de producție este în acest caz veriga cea mai slabă iar rezistența lanțului este dată de aceasta. Îmbunătățind rezistența verigii celei mai slabe, se va îmbunătăți rezistența întregului lanț. TOC fiind o tehnică ciclică, se impune reluarea punctele de identificare ale constrângerilor după fiecare parcurgere a metodei.

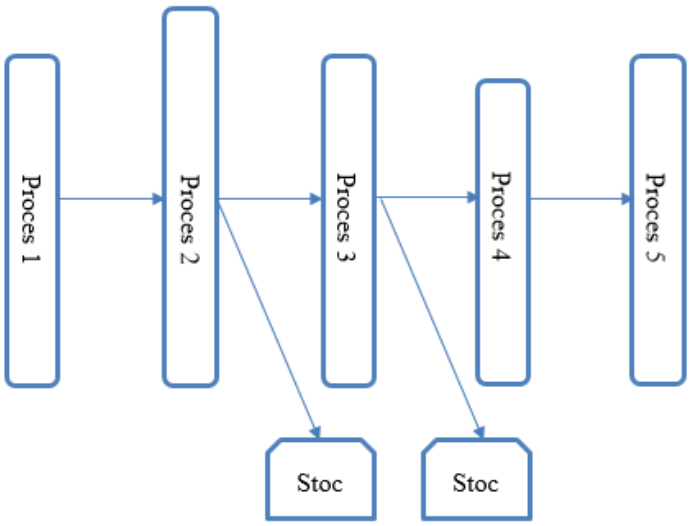

Fig. 4. 2 Lanț de producție real

Fig. 4. 3 și Fig. 4. 4 prezintă diferența dintre un lanț de producție neoptimizat și unul optimizat prin teoria constrângerilor .

În Fig. 4. 3 se prezintă cazul unui lanț de producție neoptimizat, compus din 5 procese cu următoarele capacități:

- procesul 1 (P1) are o capacitate de producție C1 de 100 de unități / zi;
- procesul 2 (P2) are o capacitate de producție C2 de 120 de unități / zi;
- procesul 3 (P3)are o capacitate de producție C3 de 80 de unități / zi;
- procesul 4 (P4) are o capacitate de producție C4 de 90 de unități / zi;
- procesul 5 (P5) are o capacitate de producție C5 de 100 de unități / zi;

Fig. 4. 3 prezintă grafic modalitatea de blocarea a capacității de producție în sub-procesele acesteia. Procesul1 având capacitatea de producție mai mică decât cel de-al doilea proces, nu va duce la pierderi datorate blocării de stocuri. Problema în lanțul de producție apare între procesele doi și trei. Procesul doi conține doi factori care duc la pierderi:

- primul factor este dimensiunea capacității procesului precedent. Procesul precedenta având o capacitate mai mică, se exclude posibilitatea ca procesul doi să lucreze la capacitate maximă.
- al doilea factor este dat de diferența de capacitate între procesul doi și trei. Diferența de capacitate intre procesele 2 și 3 va genera crearea de stocuri.

Crearea stocurilor înseamnă a bloca bani în acel stoc. Aceasta se traduce prin pierderi sau micșorarea marjei de profit prin neinvestirea sumei alocate stocului. Aceasta deoarece principalii indici de câștig în economie îl constituie inflația

și dobânda bancară. O companie are câștig numai dacă procentul profitului este mai mare decât suma dintre inflație și dobânda bancară.

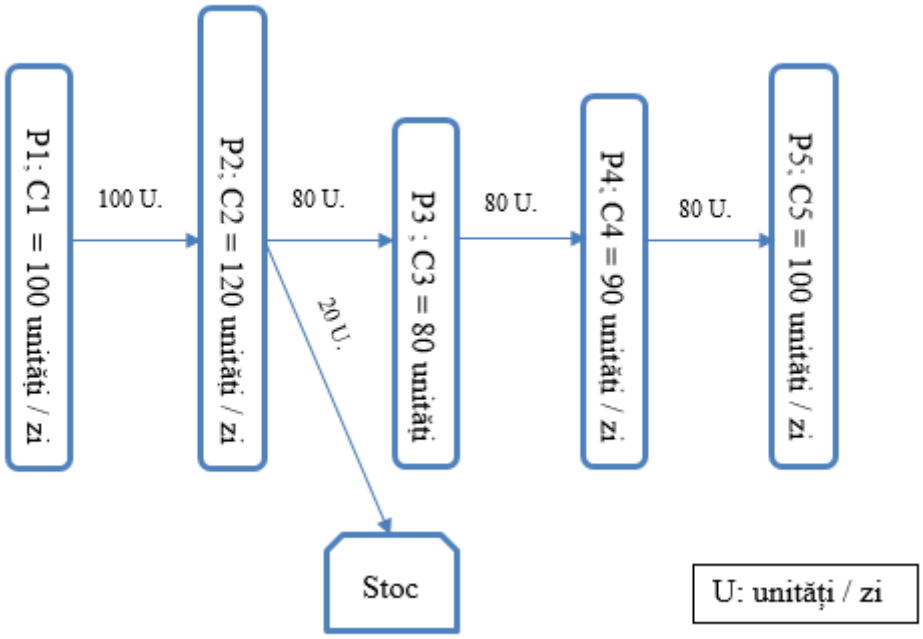

Fig. 4. 3 Lanț de producție neoptimizat.

Expresiile matematice a pierderilor provocate în fiecare proces al lanțului de producție prezentat în Fig. 4. 3 sunt:

$PI_2 = |C_2 - C_1| = |120\text{-}100| = 20$ U;

Unde PI_2 reprezintă pierderea provocată în lanțul procesului de producție până la procesul 2.

$C_2' = \min(C_1, C_2) = \min(100, 120) = 100$ U;

Unde C_2' este valoarea cea mai mică dintre capacitatea procesului 1 (C_1) și capacitatea procesului 2 (C_2).

C_2' este necesar pentru a putea calcula pierderile între doi pași consecutivi ai lanțului de producție.

$PI_3 = |C_3 - C_2'| = |80\text{-}100| = |-20| = 20$U;
$C_3' = \min(C_2', C_3) = \min(100, 80) = 80$ U;

PI3 reprezintă pierderea provocată lanțului de producție între procesul doi și procesul trei.

C_3' reprezintă minimul capacităților până la procesul 3, C_3 fiind capacitatea de producție a procesului 3.
Dacă în pasul doi, pierderea era dată de capacitatea nefolosită, în pasul trei, pierderea este definită de capacitatea superioară a pasului precedent. În acest al doilea caz, diferența de capacitate se transformă în stoc.

$PI4 = |C4 - C_3'| = |90\text{-}80| = |10| = 10U;$
$C_4' = \min(C_3', C_4) = \min(80, 90) = 80\ U;$

Unde, PI4 reprezintă pierderea provocată lanțului de producție între procesul trei și procesul patru.
C_4' reprezintă minimul capacităților până la procesul 4, C_4 fiind capacitatea de producție a procesului patru.

$PI5 = |C5 - C_4'| = |100\text{-}80| = |20| = 20U;$

Unde PI5 reprezintă pierderea provocată lanțului de producție între procesul patru și procesul cinci.

Se observă că fiecare pas al lanțului de producție produce o pierdere, ceea ce determină automat ca profitul obținut să scadă. O altă consecință ar putea fi scăderea nivelului competitivității pe piață.

Plecând de la pierderile provocate în fiecare pas de producție, se identifică posibilii pași din procese care pot fi optimizați. Fig. 4-4 prezintă grafic optimizările fiecărui pas în parte și automat al întregului lanț de producție. Capacitatea de optimizare a procesului este marcată de partea haşurată și printr-o săgeată.

Aplicarea teoriei constrângerilor descrisă de Goldratt [37] determină aceleași procese de optimizare:

- în primul pas se identifică procesul cu capacitatea cea mai scăzută. În cazul nostru este procesul 3;
- în al doilea pas se asigură exploatarea maximă a capacității resursei celei mai scăzute. În cazul din Fig. 4. 4 se asigură că procesul 3 va rula întotdeauna la capacitate maximă, adică va produce 80 de unități pe zi;
- al treilea pas al teoriei constrângerilor presupune subordonarea celorlalte procese. Procesele cu capacitate mai mare vor fi reduse la nivelul capacității constrângerilor. Resursele astfel neutilizate vor putea fi folosite în alte procese, de exemplu în elevarea constrângerii sau folosirea acestora într-un alt lanț de producție. Acest din urmă caz se aplică adesea unităților de producție care realizează mai multe tipuri de produse în aceeași fabrică;
- al patrulea pas va duce la creșterea capacității întregului lanț prin elevarea constrângerii.

Scopul unui proces de producție fiind îmbunătățirea continua a performanței sistemului, este necesar ca pașii anterior prezentați să fie parcurși în mod repetat.

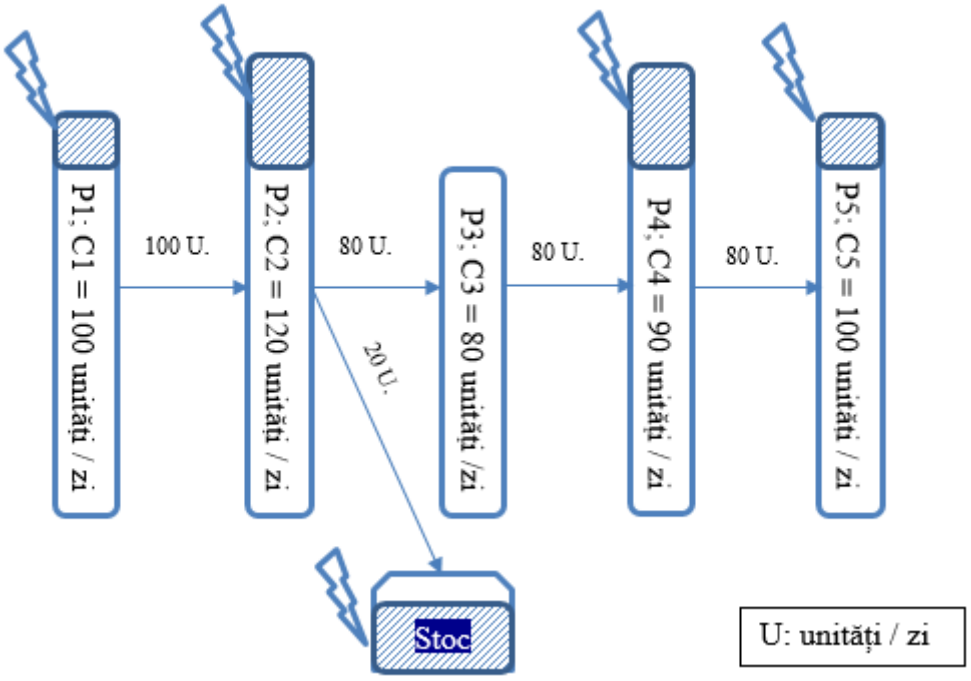

Fig. 4. 4 Capacitățile de optimizare a lanțului de producție.

Fig. 4. 5 redă imaginea lanțului de producție optimizat utilizând metoda teoriei constrângerilor. Cu scopul asigurării constrângerii sistemului la capacitate maximă, s-a luat decizia ca înaintea acesteia să fie creat un „stoc de siguranță". Rolul stocului de siguranță este de a asigura funcționarea continuă a constrângerii în cazul defectării sau opririi unui proces/utilaj aflat în înaintea constrângerii. Crearea „stocului de siguranță" determină ca procesele premergătoare constrângerii să funcționeze la capacitate ușor superioară. Altfel există posibilitatea de a se crea „stocul de siguranță" cu scopul de surmonta eventuale defecte din procesele precedente.

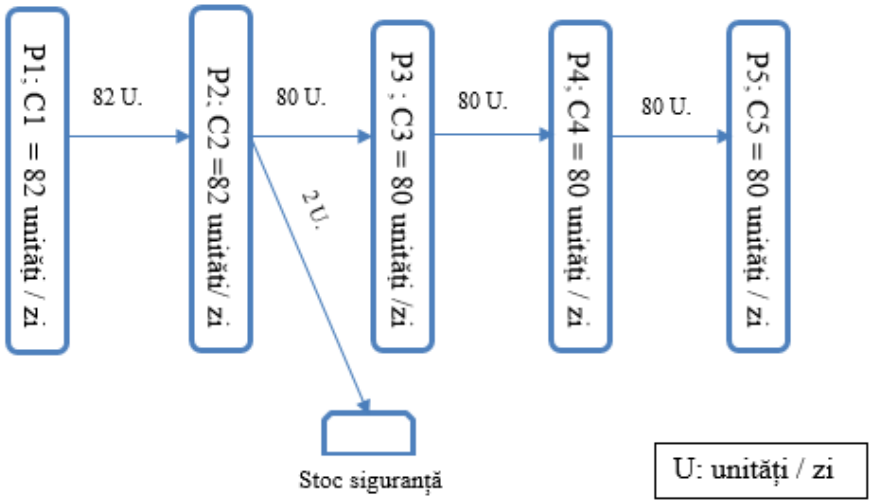

Fig. 4. 5 Lanț de producție optimizat prin teoria constrângerilor.

4.2.2 Aplicarea TOC (teoria constrângerilor) în dezvoltarea sistemelor software. Calculul drumului critic.

În dezvoltarea produselor software, respectarea specificațiilor tehnice și a timpului de predare a proiectului sunt esențiale pentru succesul produsului. Acest paragraf are ca obiectiv explicarea modalității de optimizare a proceselor folosite în proiectele software de dimensiuni mari.

Asemenea managerilor de producție, majoritatea managerilor de proiecte software încearcă să îmbunătățească fiecare proces în parte. Metode de optimizare diferă în funcție de modelul de dezvoltare utilizat. Acestea variază de la metode de revizuire a codului sursă până la utilizarea testării automate.

Deoarece sistemele software complexe presupun folosirea resurselor paralele, îmbunătățirea diferitelor sub-procese nu vor duce neapărat la optimizarea întregului proces. Optimizarea întregului lanț de dezvoltare se va putea efectua doar privind în ansamblu împărțirea resurselor și diferitele căi paralele de dezvoltare.

Conform Teoriei constrângerilor identificarea constrângerii din Fig. 4. 6 se realizează calculând drumul critic într-un proiect folosind procesele clasice. Acesta se calculează prin însumarea timpilor necesari fiecărei activități în parte. În exemplul dat, drumul critic este cel compus din activitățile A4n, unde n=[1..5], drumul critic fiind prezentat cu linii îngroșate. Se definește un proces în Fig. 4. 6 ca fiind suma activităților reprezentate pe una din liniile matricei. Resursele necesare efectuării operațiilor nu au fost luate în considerare iar optimizările legate de timpii de siguranță alocați fiecărui proces nu au fost aplicate.

Goldratt [38] explica necesitatea optimizării timpilor de siguranță alocați fiecărui proces în parte prin exploatarea și subordonarea constrângerii. Primul pas în teoria lui Goldratt constă în eliminarea timpilor de siguranță alocați fiecărei activități

(Fig. 4. 6 partea haşurată în albastru a fiecărei activităţi) şi crearea unui buffer de timp de siguranţă pentru fiecare proces. Acesta nou buffer va fi folosit în cazul unei întârzieri provocate de către una dintre activităţi aflate pe oricare dintre drumuri (Fig. 4. 7 – bufferul de proiect).

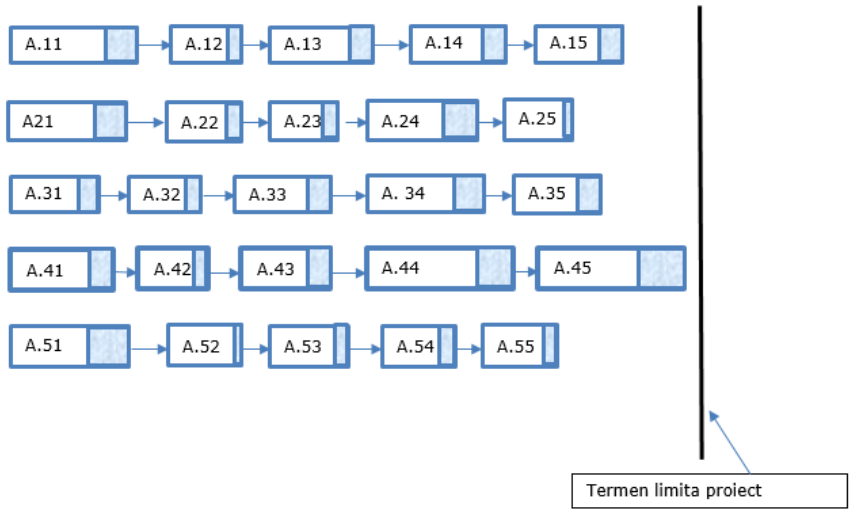

Fig. 4. 6 Drumul critic în proiectele clasice

Motivul creării unui buffer de siguranţă comun fiecărui drum este dat de tendinţa managerilor de proiect de a aloca timpi de siguranţă mult mai mari decât ar fi nevoie în realitate. Fig. 4. 7 prezintă o nouă metodă de optimizare prin crearea unui singur buffer de timp de siguranţă. Subordonarea capacităţii se realizează prin asigurarea rulării continue a acesteia la capacitate maximă, prin crearea de „buffere de alimentare". În cazul „defecţiunilor" în afara drumului critic, „bufferele de alimentare" au rolul de a asigura necesarul activităţilor de pe drumul critic (Fig.4.8).

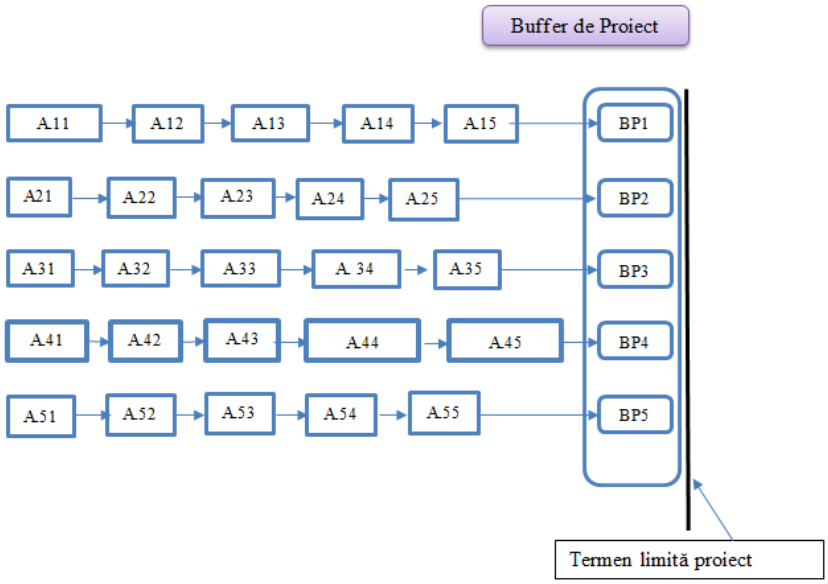

Fig. 4. 7 Optimizarea timpilor proiectului prin eliminarea timpilor de siguranță din procese.

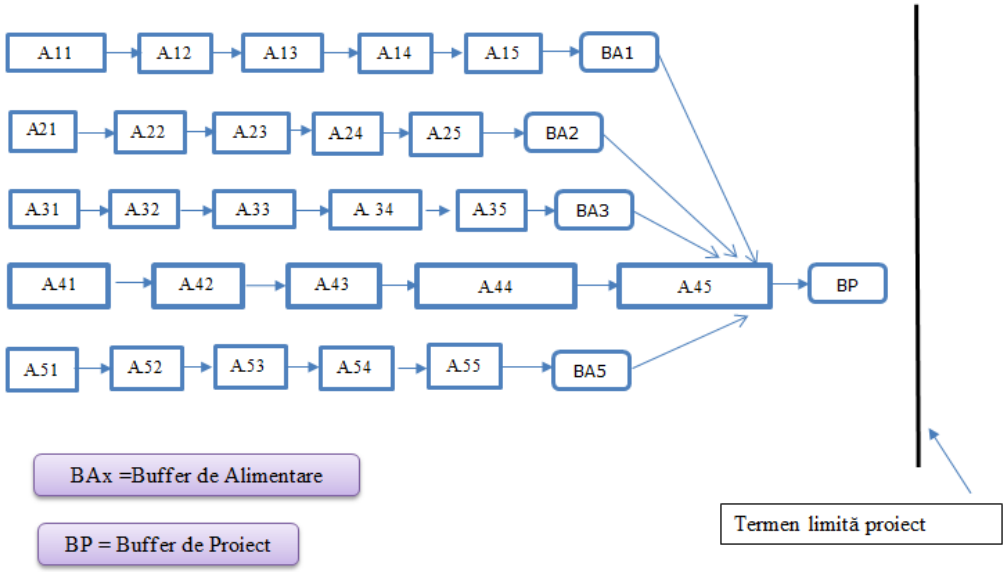

Fig. 4. 8 Subordonarea capacității lanțului de dezvoltare

Fig. 4. 9 prezintă drumul critic a unui proiect în care resursele sunt folosite paralel în mai multe activități. Goldratt [38] descria necesitatea calculului drumului critic luând în calcul toți factorii care ar putea duce către o constrângere. Drumul critic din sistemele software echivalează cu constrângerile din teoria cu același nume. Aplicarea teoriei constrângerilor asupra subproiectelor din Fig. 4. 9 va **identifica** drumul critic ca fiind **constrângerea** sistemului. Linia punctată este reprezentarea drumului critic în Fig. 4. 9.

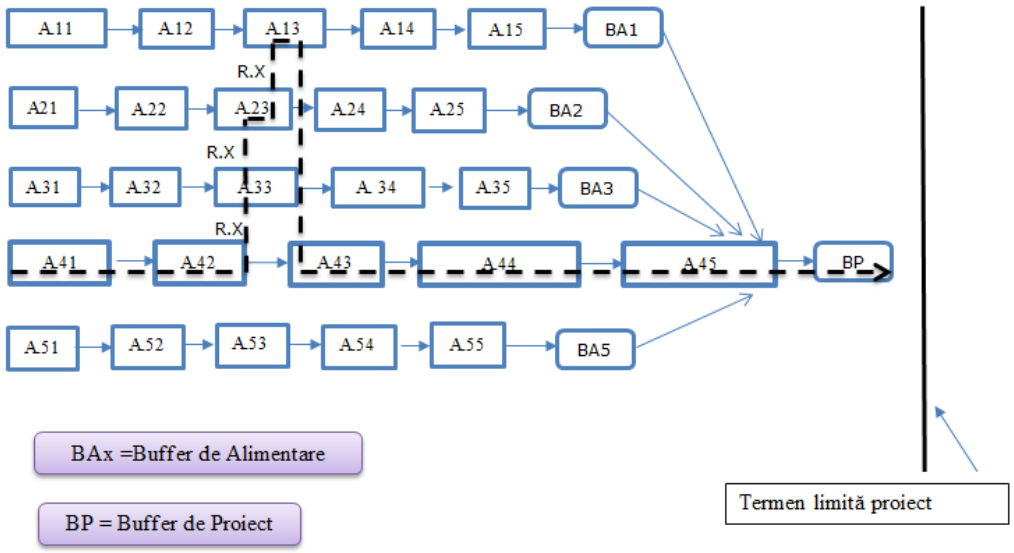

Fig. 4. 9 Drumul critic în proiecte cu resurse paralele

În continuarea aplicării primului pas din TOC trebuie găsite metode de a exploata constrângerea la capacitate maximă. În funcție de dimensiunea proiectelor software, aceasta reprezintă crearea de condiții resursei gâtuite pentru a lucra la 100% din capacitate. Acest lucru se realizează creând buffere de alimentare plasate înaintea constrângerii. Aceste buffere au rolul de a alimenta constrângerea (activitățile din drumul critic) chiar daca înaintea acesteia a survenit un blocaj neprevăzut(Fig. 4. 10).

Subordonarea capacității celorlalte pași va duce la posibilitatea de folosire a resurselor în alte activități ale proiectului sau în elevarea constrângerii. Elevarea constrângerii presupune găsirea de soluții pentru suplimentarea capacitații resursei X din Fig. 4. 10. Optimizând sub-procesele unui proiect software o singură dată, nu va asigura neapărat atingerea scopului temporal al proiectului. Acesta este motivul pentru care metodele teoriei constrângerilor vor trebui repetate, începând cu identificarea noii constrângeri a sistemului.

Procese software folosite în industria automotive sunt în continuare influențate de procesele tradiționale, procesele software fiind preluate și adaptate din procesele de producție. Flexibilitatea redusa a acestor procese este data de dimensiunea industriei, flexibilitatea proceselor fiind invers proporțională cu dimensiunea domeniilor. Folosirea modelului de dezvoltare în V cât și optimizările „împrumutate" din industriile de producție sunt datorate siguranței și stabilității date de aceste procese. Acesta este unul din motivele pentru care metodologiile agile și-au găsit cu greu aplicarea în industrii de dimensiuni mari.

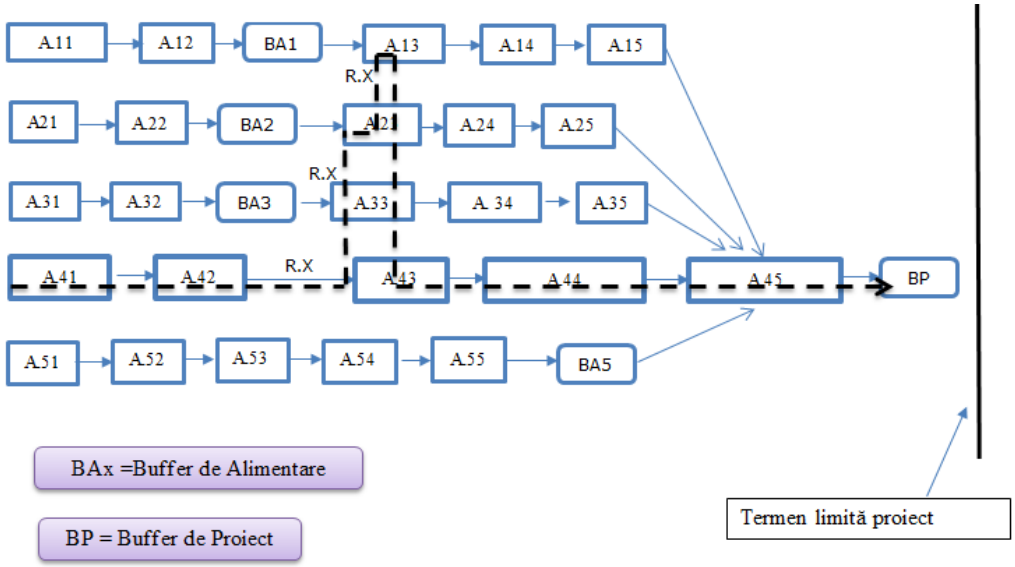

Fig. 4. 10 Poziționarea bufferelor de alimentare cu scopul exploatării constrângerii

În continuare se prezintă exemplul unui drum critic și modalitatea de aplicare a teoriei constrângerii unui proiect automotive în urma multiplelor modificări asupra cerințelor sistemului. Se consideră următorul exemplu căruia i s-au aplicat 5 modificări ale cerințelor pe parcursul evoluției acestuia (Tabel 4.1), după cum urmează:

- Modificarea modulului Trafic ca urmare a modificărilor în modulul Radio (Modificare Modul Radio =>Modificare Modul Trafic);
- Modificarea modulului de Navigație și Trafic ca urmare a modificărilor aduse în modulul de Poziționare (Modificare Modul Poziționare=> Modificare Modul Navigație și Trafic);
- Modificarea modulului Trafic ca urmare a modificărilor în modulul Navigație (Modificare Modul Navigație => Modificare Modul Trafic);
- Modificarea individuală a modulului Trafic (Modificare Modul Trafic);
- Modulul Bluetooth și Telefon ca urmare a modificărilor modulului GSM (Modificare Modul GSM => Modificare Modul Bluetooth și Modul Telefon);

O modificare a unui submodul generează modificări în alte submodule. Aceste modificări acționează anumite activități specifice modulului afectat și care sunt selectate din secvențele ciclului de viață V. De exemplu în prima coloană a matricei modificărilor de cerințe din Tabelul 4.1. sunt marcate cu 1 activitățile specifice necesare generate de modificarea cerințelor din modulul „Radio”. Durata activităților este dedusă din experiența practică a autorului ca urmare a unor modificări de complexitate medie.

Scopul este de a transpune calculul drumului critic în proiecte automotive înainte și după aplicarea teoriei constrângerilor în acest tip de proiecte. Pentru fiecare modificare a cerințelor exemplificate se identifică activitățile necesare implementării acelor modificări în module. Acestea la rândul lor vor fi înlănțuite

conform dependențelor tehnologice cu scopul aplicării algoritmului calculului drumului critic.

Tabelul 4.2 sintetizează rezultatul analizei tabelului 4.1. Drumul critic fiecărei modificări se calculează din suma timpului necesar activităților necesare implementării modificării. Se observă că drumul critic a proiectului este dat de modificările aduse modulului GSM. Metoda de calcul a drumului critic este:

DrumCritic= ∑(Analiză Cerințe Sistem Operare Bluetooth, Implementare Cerințe Sistem Operare Bluetooth, Testare Modul Sistem Operare Bluetooth, Analiză Cerințe Sistem Operare GSM, Implementare Cerințe Sistem Operare GSM, Testare Modul Sistem Operare GSM, Analiză Cerințe Aplicație, Implementare Cerințe Aplicație Telefon, Testare Modul Aplicație Telefon, Analiză Cerințe HMI , Implementare Cerințe HMI Telefon, Testare Modul HMI, Testare sistem);

Numărul ciclurilor în V necesare implementării modificărilor este dat de raportul între durata timpul necesar activităților de implementare cu numărul zilelor necesare rulării unui ciclu (5).

Aplicarea teoriei constrângerii asupra exemplului anterior, prezentat și în fig. 4.11 identifică constrângerea ca fiind drumul critic. În fig. 4.11 drumul critic este reprezentat de linia roșie și constă din înlănțuirea de activități cu durata cea mai mare. Exploatarea constrângerii presupune rularea continuă a activităților de pe drumul critic la 100% din capacitate, drumurile necritice neinfluențând termenul limită al proiectului în mod direct. În acest scop se vor tăia toți timpii de siguranță alocați activităților din drumului critic, știindu-se tendința coordonatorilor de proiect de a aloca un timp de siguranță mai mare decât cel strict operațional pentru fiecare activitate. Timpii de siguranță eliminați din fiecare activitate vor fi însumați într-un buffer de timp al proiectului, mai concret un timp de siguranță pentru drumul critic în condițiile în care durata acestuia a fost scurtată datorită scurtării marjelor de siguranță.

Tabel 4. 1 Durata implementării modificărilor complexe în modulele sistemului

	Durata activităților		Matricea modificărilor de cerințe				
Activități	**Durata inițială (în zile)**	**Durata după eliminare timpi de siguranță (în zile)**	Modificare Modul Radio =>Modificare Modul Trafic	Modificare Modul Poziționare=> Modificare Modul Navigație și Trafic	Modificare Modul Navigație => Modificare Modul Trafic	Modificare Modul Trafic	Modificare Modul GSM => Modificare Modul Bluetooth și Modul Telefon
Analiză Cerințe Aplicație Radio	1	1	1	0	0	0	0
Implementare Cerințe Aplicație Radio	3	2	1	0	0	0	0
Testare Modul Aplicație Radio	2	2	1	0	0	0	0
Analiză Cerințe Sistem Operare Poziționare	1	1	0	1	0	0	0
Implementare Cerințe Sistem Operare Poziționare	4	2	0	1	0	0	0
Testare Modul Sistem Operare Poziționare	2	2	0	1	0	0	0
Analiză Cerințe Aplicație Navigație	1	1	0	1	1	0	0
Implementare Cerințe Aplicație Navigație	2	1	0	1	1	0	0
Testare Modul Aplicație Navigație	1,5	1	0	1	1	0	0
Analiză Cerințe HMI Navigație	1	1	0	1	1	0	0
Implementare Cerințe HMI Navigație	1,5	1	0	1	1	0	0
Testare Modul HMI Navigație	2	1	0	1	1	0	0
Analiză Cerințe Aplicație Trafic	1	1	1	1	1	1	0
Implementare Cerințe Aplicație Trafic	3	2	1	1	1	1	0
Testare Modul Aplicație Trafic	1,5	1	1	1	1	1	0

Continuare Tabel 4.1

	Durata activităţilor		Matricea modificărilor de cerinţe				
Activităţi	**Durata iniţială (în zile)**	**Durata după eliminare timpi de siguranţă (în zile)**	Modificare Modul Radio =>Modificare Modul Trafic	Modificare Modul Poziţionare=> Modificare Modul Navigaţie şi Trafic	Modificare Modul Navigaţie => Modificare Modul Trafic	Modificare Modul Trafic	Modificare Modul GSM => Modificare Modul Bluetooth şi Modul Telefon
Analiză Cerinţe Sistem Operare Bluetooth	1	1	0	0	0	0	1
Implementare Cerinţe Sistem Operare Bluetooth	4	2	0	0	0	0	1
Testare Modul Sistem Operare Bluetooth	2	2	0	0	0	0	1
Analiză Cerinţe Sistem Operare GSM	1	1	0	0	0	0	1
Implementare Cerinţe Sistem Operare GSM	3	2	0	0	0	0	1
Testare Modul Sistem Operare GSM	2	1	0	0	0	0	1
Analiză Cerinţe Aplicaţie Telefon	1	1	0	0	0	0	1
Implementare Cerinţe Aplicaţie Telefon	4	2	0	0	0	0	1
Testare Modul Aplicaţie Telefon	2	1,5	0	0	0	0	1
Analiză Cerinţe HMI Telefon	1	1	0	0	0	0	1
Implementare Cerinţe HMI Telefon	3	2	0	0	0	0	1
Testare Modul HMI Telefon	2	1	0	0	0	0	1
Testare sistem	3	2	1	1	1	1	1

Tabel 4. 2 Timpii necesari parcurgerii drumurilor critice și necritice.

Module afectate de modificare	Modificare Modul Radio =>Modificare Modul Trafic	Modificare Modul Poziționare=> Modificare Modul Navigație și Trafic	Modificare Modul Navigație => Modificare Modul Trafic	Modificare Modul Trafic	Modificare Modul GSM => Modificare Modul Bluetooth și Modul Telefon
Durata drumului (în zile)	14,5	24,5	17,5	8,5	29
Cicluri de dezvolatare în V	2,9	4,9	3,5	1,7	5,8

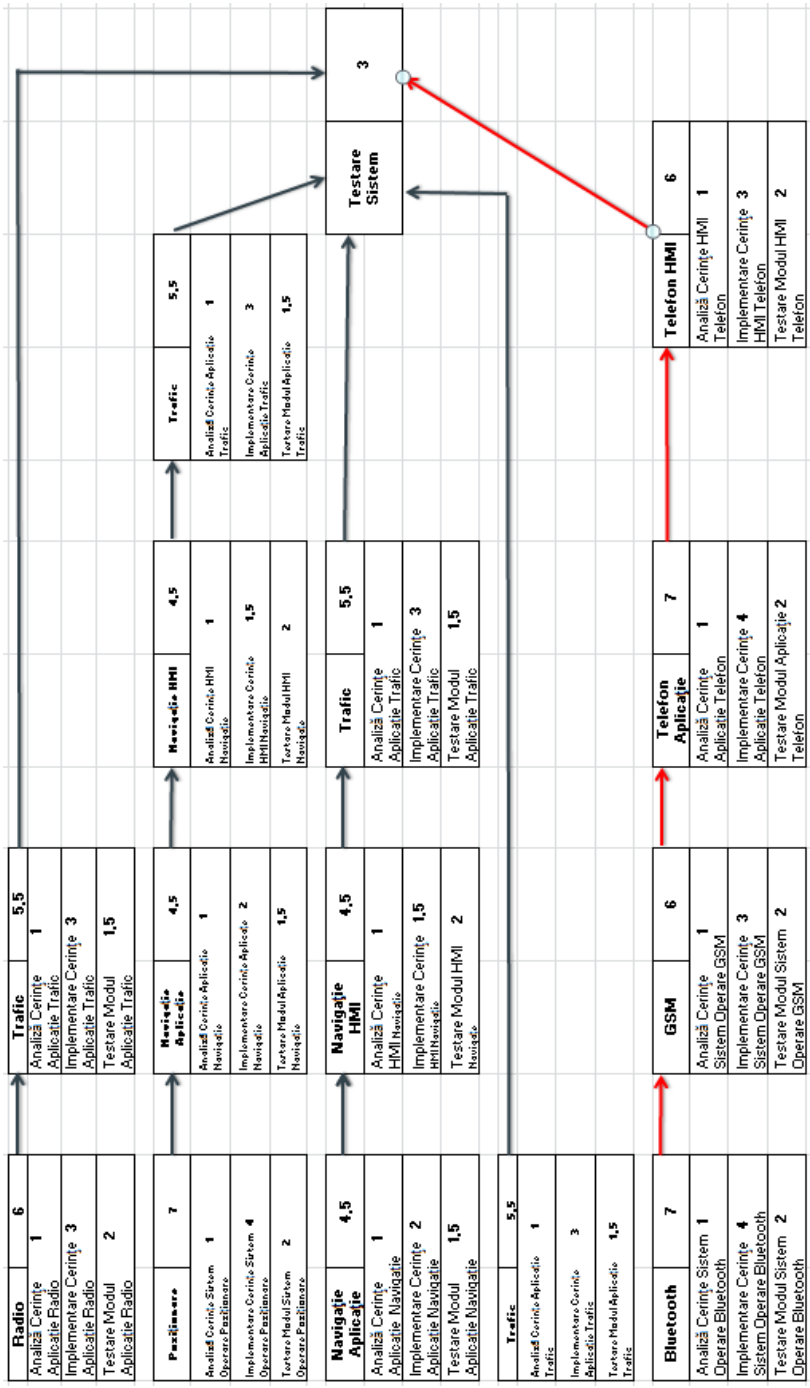

Fig. 4. 11 Reprezentarea drumului critic într-un proiect automotive în urma modificărilor cerințelor – Durate cu marjă de siguranță

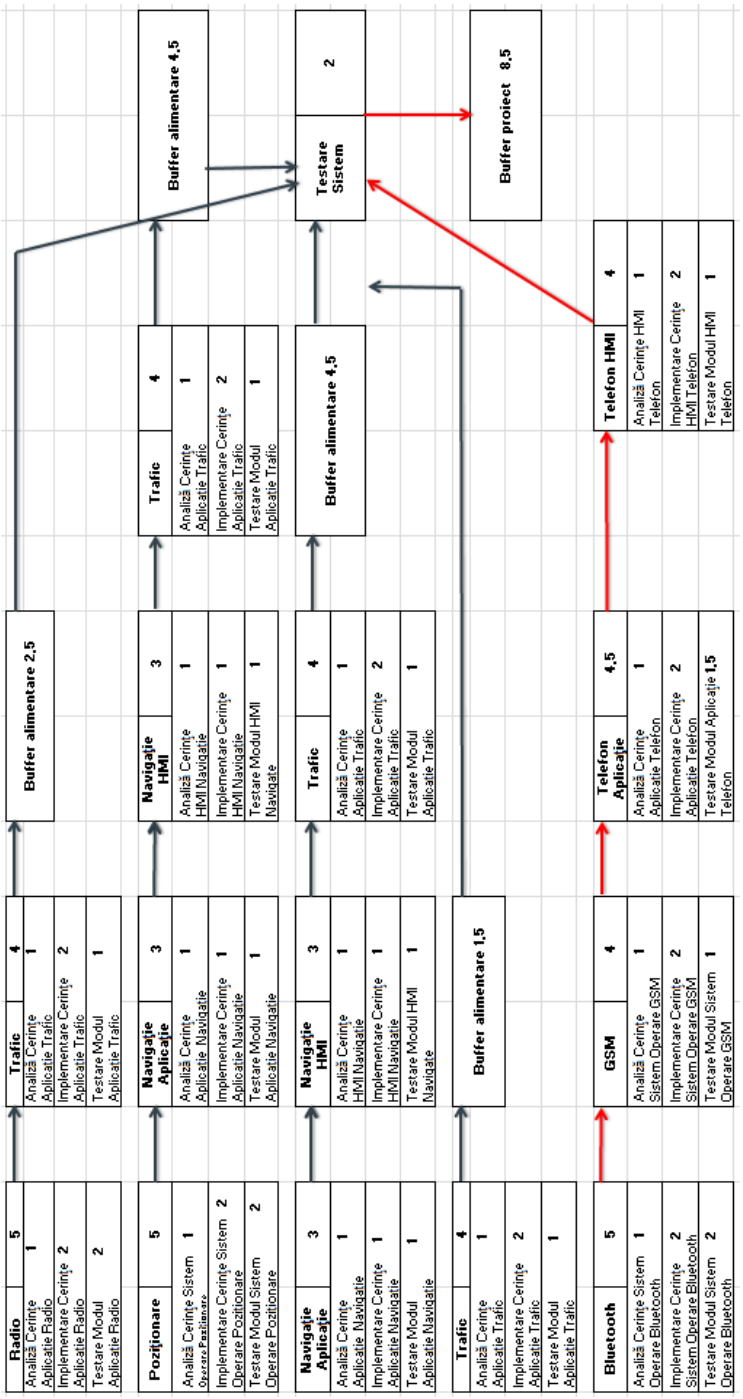

Fig. 4. 12 Reprezentarea drumului critic într-un proiect automotive în urma modificărilor cerinţelor şi eliminării timpilor de siguranţă – durate strict operaţionale fără marjă de siguranţă.

Subordonarea presupune reluarea pasului prezentat anterior pentru drumurile considerate necritice. Tabelul 4.3 prezintă timpul necesar noului drum critic în urma eliminării timpilor de siguranță. Rezultatul este calculat analog tabelului 4.2 prin însumarea activităților desfășurate pentru implementarea modificării. Numărul ciclurilor în V necesare implementării unei modificări este calculat din raportul dintre durata fiecărui drum și durata în zile parcurgerii unui ciclu în V, în cazul nostru specific durata parcurgerii unui ciclu este de cinci zile.

Tabel 4. 3 Timpii necesari parcurgerii drumurilor critice și necritice în urma eliminării timpilor de siguranță.

Module afectate de modificare	Modificare Modul Radio =>Modificare Modul Trafic	Modificare Modul Poziționare=> Modificare Modul Navigație și Trafic	Modificare Modul Navigație => Modificare Modul Trafic	Modificare Modul Trafic	Modificare Modul GSM => Modificare Modul Bluetooth și Modul Telefon
Durata drumului după eliminare timpi de siguranță(în zile)	11	17	12	6	19,5
Cicluri de dezvolatare în V	2,2	3,4	2,4	1,2	3,9

Fig. 4. 12 arată în mod grafic pasul de subordonare a drumurilor necritice proiectului prin crearea de buffere de alimentare. Metoda calculului drumului critic are drept scop reducerea timpului de rulare a proiectului, coordonatorii de proiect urmărind progresul proiectului prin intermediul bufferelor create ca urmare a eliminării timpilor de siguranță alocați fiecărei activități.

Deoarece teoria constrângerii trebuie aplicată pe toată durata proiectului, se va relua pasul de identificare a constrângerii, luând în calcul resursele necesare implementării proiectului. În proiectele automotive a sistemelor de radio navigație este un fapt normal ca echipa care implementează modulul trafic să fie aceeași și pentru aplicația sau interfața om-mașină a modulelor de navigație.

Se identifică astfel lanțul critic ca fiind drumul compus din activitățile necesare implementării modulelor de navigație de către resursa „X" (Fig. 4. 13). Noul lanț critic este prezentat de linia punctată și este format din activitățile interdependente ale drumurilor critice și necritice ale proiectului. Durata lanțului critic va fi întotdeauna mai mare sau cel mult egal cu drumul critic. Dependența lanțului este dată de necesitatea „așteptării" anumitor activități până la eliberarea resursei „X". Resursa considerată în calculul lanțului critic fiind considerată o nouă constrângere a proiectului.

Exploatarea și elevarea presupune asigurarea rulării activităților din lanțul critic la capacitate maximă. Activitățile fiind planificate realist-optimist, este necesar crearea unui buffer de acoperire a eventualelor probleme apărute. Acesta este bufferul proiectului. Subordonarea impune repoziționarea bufferelor de alimentare. Poziționarea bufferelor de alimentare în fața activităților din lanțul critic va permite rularea constrângerii la capacitate maximă, chiar dacă pe unul din drumurile necritice apare o „defecțiune" (Fig. 4. 14).

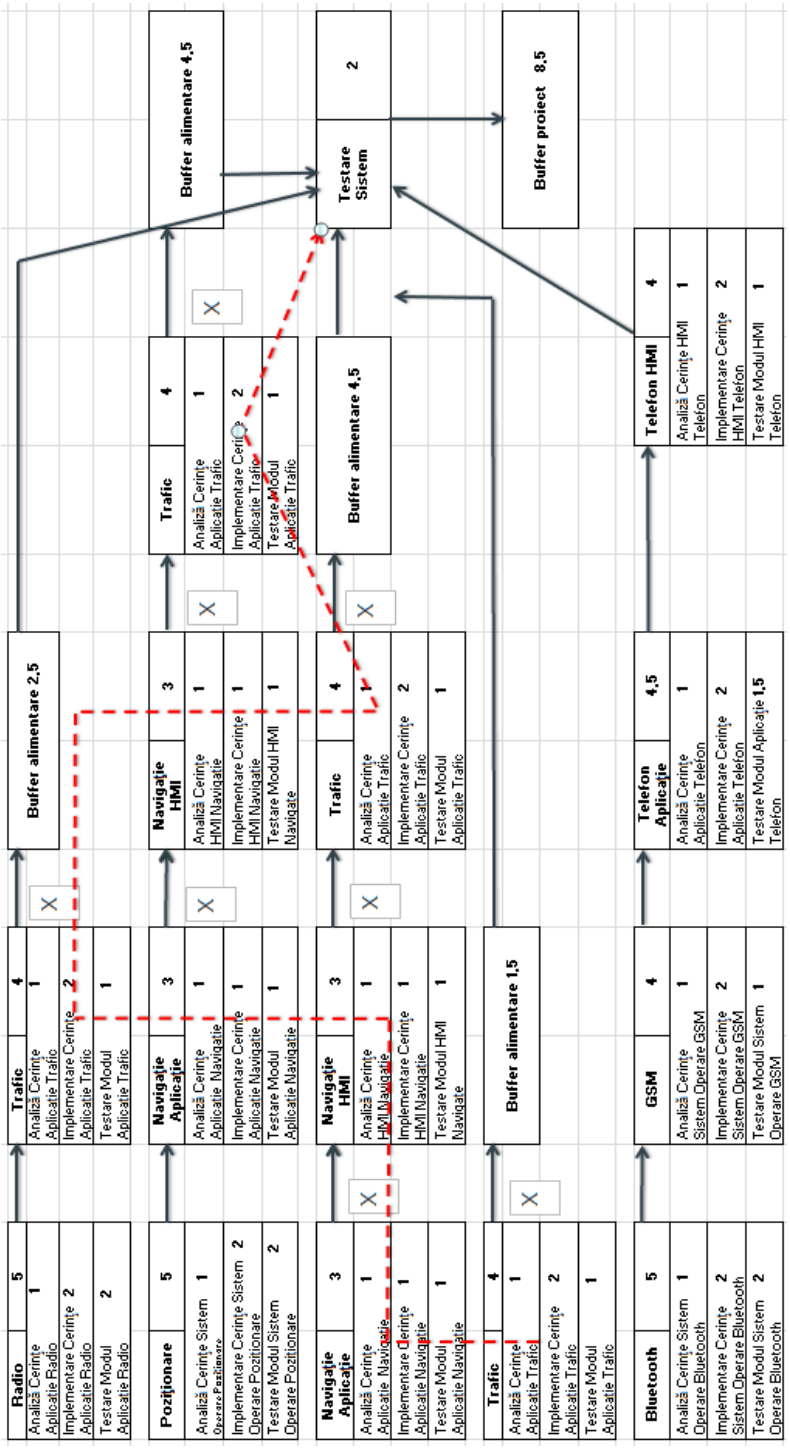

Fig. 4. 13 Reprezentarea lanțului critic într-un proiect automotive

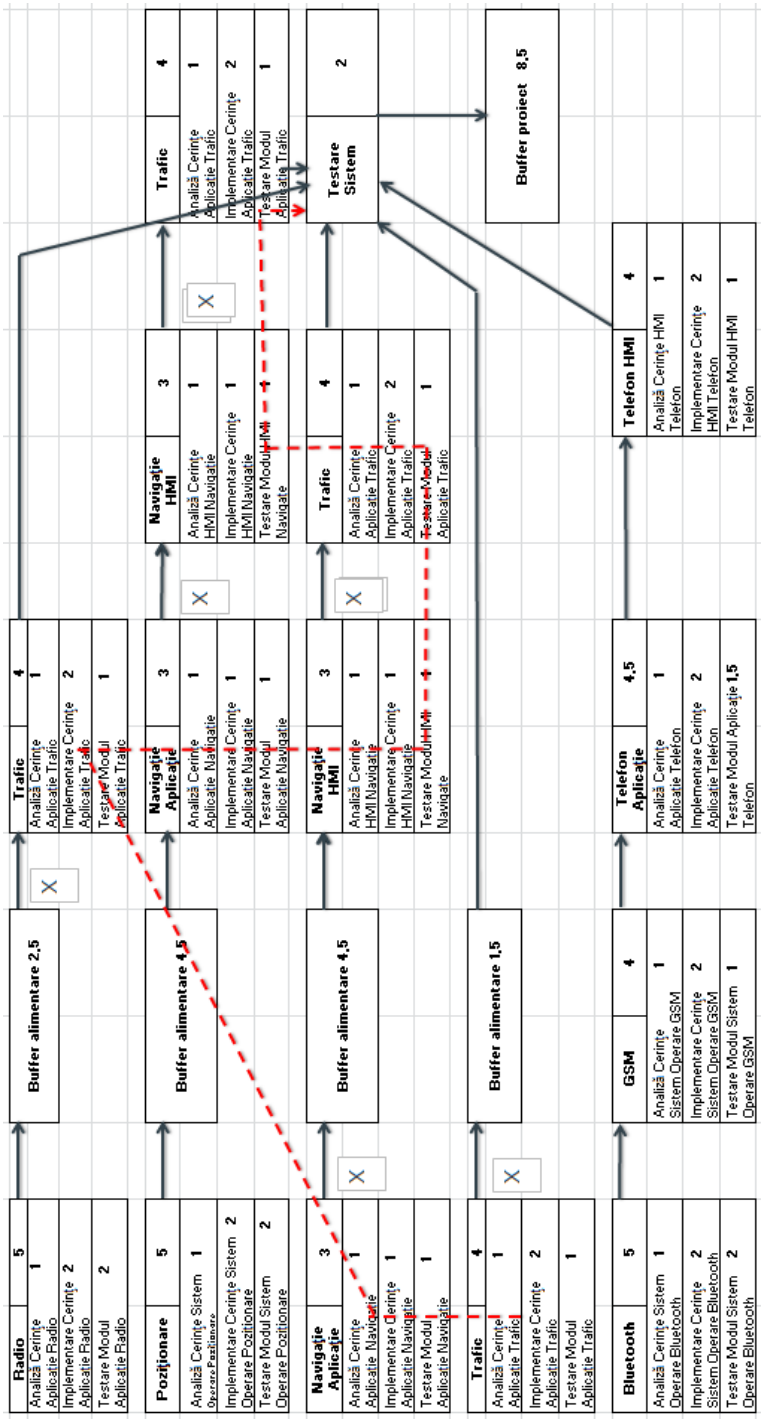

Fig. 4. 14 Subordonarea activităților la constrângerea lanțului critic într-un proiect automotive

Analizarea Fig. 4. 14 duce la concluzia că modificările aduse proiectelor vor trebui planificate și prioritizate în funcție de necesitățile și scopul proiectelor. Tabel 4. 4 prezintă diferența de timp între drumul critic și lanțul critic în cazul unui proiect automotive care este supus la cinci modificări ale specificațiilor prezentate anterior.

Tabel 4. 4 Comparație între durata necesară implementării modificărilor

Durata drumului critic (în zile)	**29**
Durata drumului critic **după eliminare timpi de siguranţă(în zile)**	19,5
Durata lanţului critic (în zile)	38,5

În concluzie, se observă că în urma aplicării teoriei constrângerilor în proiectele automotive se economisește timp, economie care însă nu permite implementarea completă a tuturor modificărilor de cerințe sau a funcțiilor planificate inițial. Modificarea cerințelor în timpul implementării unui sistem complex de radio navigație creează nevoia de implementare adițională de noi metodologii de economisire de timp.

4.3 Conceperea unui model de dezvoltare dinamic a sistemelor complexe automotive (2JCS)

4.3.1 Factori decizionali care duc la implementarea cerințelor modificate

Din literatura de specialitate reiese că schimbările aduse cerințelor proiectelor se vor modifica în timpul dezvoltării proiectelor iar obiectivul principal al proiectelor sistemelor de radio-navigație este respectarea factorului temporal al proiectelor. Acest obiectiv reiese din dorința clientului proiectului de a respecta etapele temporale ale proiectului, ceea ce a dus la urmărirea detaliată și riguroasa a etapelor proiectului [5- Huțanu].

Urmărirea riguroasă a etapelor proiectului determină furnizorul să refuze implementarea cerințelor noi pe parcursul desfășurării proiectului. Unul din factorii care determină necesitatea de implementare a cerințelor modificate este crearea de noi produse tehnologice. Aceasta contravine de cele mai multe ori obiectivului temporal al proiectului, ceea ce generează un conflict intre clientul și furnizorul proiectului.

Schimbarea cerințelor este generată în principal de faptul că ciclul de viață a noilor tehnologii nu mai corespunde curbei clasice [38]. Specificațiile sistemelor de radio-navigație conțin cerințe care nu vor mai fi actuale în momentul lansării produsului pe piață. Motivul este dat de durata mare a ciclului de viață a proiectelor sistemelor de radio-navigație.

Schimbarea cerințelor este generată în principal de:

- diferența duratei ciclurilor de viață ale sistemelor de radio-navigație și a componentelor periferice acestora precum telefoanele mobile;
- specificații neclare [6- Huțanu];

- specificații incomplete [6- Huțanu];
- erori generate de implementarea unor cerințe care contracarează cu alte cerințe[6- Huțanu];

Factorii care influențează decizia schimbării din punctul de vedere al clientului proiectului sunt:

- cerințele pieței;
- cerințe implementate greșit datorită descrierii insuficiente;
- erori software sau hardware;
- analiza impactului asupra pieței;
- analiza riscului:
  - o numărul liniilor de cod care trebuie modificate;
  - o numărul componentelor software care vor fi modificate;
  - o timpul necesar implementării schimbării;
- timpul necesar implementării schimbării, inclusiv validarea implementării;

Factorii care influențează decizia furnizorului de a implementa noile cerințe sunt:

- importanța părților interesate;
- structura proiectului;
- planul temporal al proiectului;

Riscul implementării cerințelor modificate diferă în funcție de fazele proiectului; proiectele prezintă un risc redus în stadiile timpurii, schimbările cerințelor într-o fază târzie a proiectului crescând riscul eșecului proiectului. Acest risc este amplificat de factorul uman. Importanța factorului uman în proiecte este subliniat de către Reiss [103] deoarece „proiectul este o activitate umană care îndeplinește un obiectiv corelat la o scală temporală”.

Figura 4-9 prezintă structura unui proiect de dezvoltare software utilizând modelul de dezvoltare în V precum și influența schimbărilor asupra structurii proiectului. Premisa acestei figuri este că modificările aduse specificațiilor să poată fi implementate fără a afecta planificarea temporală inițială a proiectului.

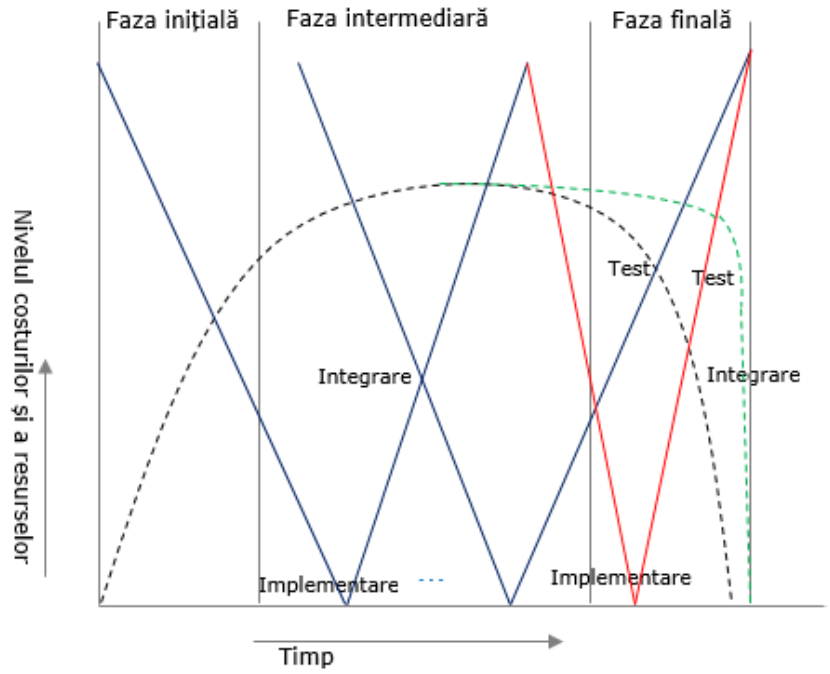

Fig. 4. 15 Adaptare după PMBOK: Dezvoltarea costurilor şi a resurselor de-a lungul dezvoltării proiectelor [6-Huțanu].

Linia neagră punctată reprezintă dimensiunea echipei proiectului în diferitele faze ale acestuia înainte însă de a efectua schimbări ale cerințelor. În faza inițială dimensiunea echipei proiectului este mică, crește către faza intermediară și urmează să scadă în faza finală a proiectului.

În momentul acceptării implementării schimbării într-o fază târzie, proiectul va trebui să implementeze conform teoriei a cel puțin încă unui ciclu, fapt reprezentat de un ciclul în V suplimentar desenat cu roșu în Fig. 4. 15. Implementarea ciclului suplimentar duce la menținerea echipei proiectului pe toată durata implementării cerințelor noi pentru a identifica eventuale erori sau regresii care pot apărea în sistemul software.

Linia punctată verde din Fig. 4. 15 prezintă noua dimensiune a echipei proiectului în urma implementării schimbărilor. Deoarece data de predare a proiectului nu poate fi modificată, curba de reducere a dimensiunii echipei de proiect va fi abruptă.

4.3.2 Măsuri de contracarare a riscului schimbării cerințelor în proiectele sistemelor de radio- navigație

Riscurile generate de schimbarea cerințelor proiectelor software de dimensiuni mari sau a proiectelor sistemelor de radio-navigație necesită dezvoltarea unor masuri de control al riscului. Orice modificare adusă sistemului va fi analizată

urmându-se a se realiza un studiu al impactului modificării asupra riscului implementării cât și a impactului asupra competitivității produsului pe piață.

Deoarece proiectele rareori se desfășoară conform planului proiectului [102], schimbările cerințelor duc la necesitatea creării unei echipe de control a schimbării.

În proiectele automotive, rolul procesului de control al cerințelor este de a asigura că schimbările cerințelor sunt coordonate, urmărite și controlate [76]. Totodată eventualele modificări ale cerințelor trebuie documentate pentru a identifica progresul acestora precum și pentru a analiza riscul modificării de către toate modulele afectate [76].

Rolul echipei de control a schimbării este de a analiza fiecare schimbare a cerințelor precum și efectele implementării acesteia:

- „Gestionarea cerințelor acceptate în momentul apariției acestora prin reglarea fluxului de schimbări"[102];
- „Controlul și actualizarea conținutului, bugetului și a planului proiectului prin coordonarea modificărilor pe întreaga durată a proiectului" [95];
- Asigurarea că procesul schimbării cerințelor începe imediat după faza de nominalizare a furnizorului. Versiunea cerințelor transmise de către departamentul de achiziții reprezintă referința din punctul de vedere al cerințelor [108];
- Închiderea proiectului reprezintă totodată și finalizarea procesului de control al schimbărilor [108];
- Prioritizarea schimbărilor în funcție de stadiul proiectului;

Numărul schimbărilor de cerințe în proiectele sistemelor de radio-navigație este direct proporțională cu [108]:

- Organizarea de ședințe de validare a conceptelor între furnizor și OEM(Original Equipment Manufacturer);
- Organizarea de test drive-uri la care participă managerii OEM-urilor;
- Organizarea de test drive-uri la care participă departamentele de testare și de calitate ale OEM-urilor;

Pentru ca măsura creării echipei de control să aibă efect, aceasta trebuie să fie condusă de către un coordonator al modificărilor și erorilor. Acesta are rolul de a asigura că toți participanții la proiect vor respecta procesul schimbării cerințelor. Scopul instalării echipei de control este minimizarea riscului proiectului prin analiza factorilor care determină prioritatea schimbării.

Pe baza experienței și a expertizei autorului pe baza factorul timp, cost și calitate, autorul a realizat o formulă matematică care oferă o recomandare în privința acceptării implementării unei modificări a cerințelor.

Pornind de la cei 3 factori tehnici (numărul liniilor de cod, numărul modulelor ce urmează a fi modificate pentru a implementa noua cerință și planul temporal al proiectului), aceștia trebuie să fie analizați în ansamblu. Pe lângă analiza celor trei factori, autorul a identificat din experiența personală mai mulți factori care ajută la luarea unei decizii legate de acceptarea schimbării:

- **C**osturile orare generate de schimbarea cerințelor pentru fiecare resursă a furnizorului (c.s.).
- **N**umărul resurselor necesare furnizorului pentru a implementa noua cerință (n.M.).

- **C**osturile orare generate de schimbarea cerințelor pentru fiecare resursă a **c**lientului (c.h.c).
- **T**impul necesar fazei de integrare (t.i.p.).
- **T**impul necesar fazei de testare a criteriilor de acceptanță (t.t.p.c).
- **N**umărul de resurse necesare clientului pentru a testa cerința schimbată (n.M.T).
- **C**osturile generate pentru pregătirea fazei de testare a criteriilor de acceptanță (c.P).
- **I**mportanța schimbării cerinței pentru piață (M.S.).
- **E**valuarea riscului (R.A.).

În urma evaluării riscului, acesta poate lua valori în intervalul [0..1], unde:

- Valoarea minimă înseamnă că riscul implementării cerinței schimbate este zero.
- Valoarea maximă înseamnă că riscul implementării cerinței schimbate este foarte mare.

Tabel 4. 5 Probabilitatea de apariție.

Probabilitatea de apariție	Procentaj
Foarte scăzută	< 5%
Scăzută	>= 5%; <25%
Medie	>=25%; <50%
Ridicată	>=50%; <90%
Foarte ridicată	>=90%

Evaluarea riscului implementării noii cerințe este determinat în principal de următorii factori folosind metoda HAZOP (Hazard și Operabilitate) [80]:

- Numărul liniilor de cod care trebuie modificate pentru a implementa noua cerință;
- Numărul componentelor software care vor fi modificate în urma implementării noii cerințe;

Tabel 4. 6 Scala severității.[6-Huţanu]

Severitate	Valoare
Scăzută	6
Medie	4
Ridicată	2
Foarte ridicată	1

HAZOP este o tehnică de analiză utilizată pentru studiul hazardului cât și a problemelor de operabilitate, analizând efectele oricărei deviații începând cu condițiile definite inițial [68].

Echipa de dezvoltare software va evalua fiecare modificare analizând scala severității (Tabel 4. 6) și probabilitatea de apariție a riscului (Tabel 4.5). Valoarea scalei severității este dată de valoarea „stricăciunilor" care pot rezulta în urma apariției hazardului. Analiza combinată a celor doi factori va genera valoarea riscului (R.A.). Valoarea riscului este prezentata sub forma unei matrice a riscului în Fig. 4. 16.

Probabilitatea de apariție	**Severitate**			
	Scăzută	Medie	Ridicată	Foarte ridicată
Foarte scăzută	0	0,2	0,3	0,4
Scăzută	0,2	0,3	0,4	0,6
Medie	0,3	0,3	0,6	0,8
Ridicată	0,4	0,6	0,8	0,9
Foarte ridicată	0,6	0,8	0,9	1

Fig. 4. 16 Matricea riscului [6-Hutanu]

Importanța schimbării cerinței pentru piață este expresia avantajului adus proiectului implementând cererea de schimbare. Valoarea importanței pentru piață a schimbării este dată de departamentul de marketing a OEM-ului.

Raportul matematic generat de factorii prezentați mai sus este:

$$\rho = \frac{t.Costs * R.A.}{M.S.} * 10 \qquad (1)$$

unde

$$t.Costs = n.M. * c.s. * t.i.p. + n.M.T. * c.h.c. * t.t.p.c. + c.P \qquad (2)$$

Formula (1) prezintă raportul între cost, risc și importanța schimbării pentru piață. Rezultatul raportului oferă o indicație asupra avantajului implementării cererii de schimbare.

Acest raport (1) nu generează răspunsul la întrebarea dacă o nouă cerință de sistem trebuie să fie implementată sau nu, ci oferă o recomandare a implementării. Cu cât rezultatul raportului este mai mic sau tinde către zero, cu atât această schimbare va aduce un avantaj mai mare proiectului.

Dacă rezultatul raportului (1) este 1 (unu), atunci avantajul modificării cerinței este egală cu costurile generate de implementarea schimbării. Orice rezultat al raportului care depășește valoarea 1 (unu) indică o pierdere adusă proiectului.

Acest raport este un instrument de recomandare a efectuării schimbării. Aceasta deoarece există cazuri în care o anumită cerință respectiv cazuri de utilizare vor aduce prestigiu rezultatului proiectului.

4.3.2.1 Analiza și elaborarea de noi modele de dezvoltare bazate pe necesitățile implementării noilor tehnologii în industria automotive (2JCS)

Prin gestionarea și coordonarea fluxul de schimbări de cerințe, riscul de eșec al proiectului scade, depinzând în mare măsură de numărul de schimbări ale cerințelor acceptate și de complexitatea acestora. Adițional, pe lângă condițiile de performantă care trebuie îndeplinite pentru a menține sau câștiga cota de piață, companiile trebuie să:

- dezvolte produse de calitate superioară;
- găsească soluții pentru a reduce timpul necesar dezvoltării și implementării noilor produse;

Acest nou cadru a dus la necesitatea studierii a noi tehnici mai eficiente a managementului de proiect.

Elaborarea unui model de dezvoltare care să corespundă industriei automotive și care să fie dinamic și tolerant la schimbări duce la necesitatea încadrării schimbărilor în mai multe categorii:

- schimbări care prezintă un risc mic de implementare. Aceste schimbări presupun modificarea unui număr redus de linii de cod și unui singur modul;
- schimbări care prezintă un risc mediu de implementare. Aceste schimbări presupun modificarea mai multor module, modificările aduse fiind însă clar definite;
- schimbări care prezintă un risc mare de implementare. Aceste schimbări presupun modificări ale arhitecturii sistemului sau a diferitelor module. În acest caz neputând-se stabili reacția sistemului la modificare;

Din structura ciclului de viață al proiectului se observă că modificările aduse caietului de sarcini pot surveni numai în urma nominalizării furnizorului. Aceasta deoarece versiunea inițială a caietului de sarcini determină câștigătorul proiectului în funcție de costurile inițiale și calendarul proiectului.

În organizarea actuală a proceselor automotive, schimbarea cerințelor va duce cu siguranță la întârzieri în planul proiectului. Datorită rigidității modelelor de implementare tradiționale folosite în dezvoltarea proiectelor de radio-navigație, orice modificare a cerințelor va duce la întârzieri ale proiectelor.

4.3.2.2 Efectul schimbării cerințelor într-un model clasic de dezvoltare automotive și analiza întârzierilor utilizând metoda drumului critic

Fig. 4.17 prezintă impactul asupra calendarului proiectului în urma unei schimbări a cerințelor în cazul în care proiectul se află în faza de proiectare a arhitecturii (Fig. 4.17a*) sistemului. Ciclul de viață va trebui reluat de la prima fază, reprezentat prin ciclul în V desenat cu albastru din fig. 4.17.fie credibil

Modelul de dezvoltare în V nu oferă flexibilitatea de a amortiza efortul generat de modificarea cerințelor. Întârzierea provocată este reprezentată de timpul scurs între specificarea cerințelor și momentul acceptării noii cerințe de către

furnizor, în exemplul dat, faza de proiectare a arhitecturii sistemului. Întârzierea provocată proiectului de noile cerințe este dependentă de momentul în care este comunicată decizia de implementare a acestora precum și de complexitatea schimbării. Schimbările cu risc de implementare ridicat vor produce întârzieri semnificative proiectelor.

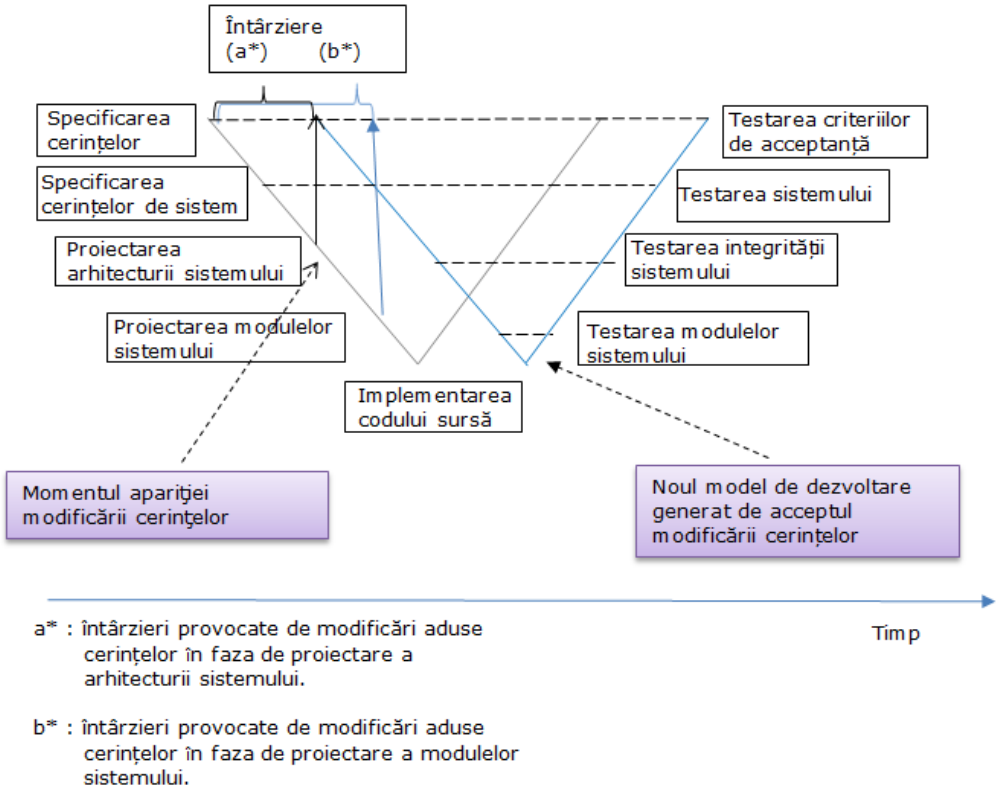

Fig. 4. 17 Efectele schimbării cerințelor asupra modelului în V

În urma aplicării teoriei constrângerii se identifică noul drum critic al proiectului ca fiind drumul cel mai lung calculat până la realizarea și testarea cerințelor. În Fig. 4. 18 drumul critic este reprezentat prin linia punctată roșie.

Prin modificarea cerințelor se decalează calendarul proiectului. Conform experienței autorului decalajul se datorează în principal rigidității ciclului de viață folosit în dezvoltarea proiectului precum și a multitudinii de probleme care apar în timpul dezvoltării sistemelor de radio-navigație. Din aceleași motive recuperarea timpului compromis de schimbarea cerințelor nu va putea fi realizată până la finalul proiectului.

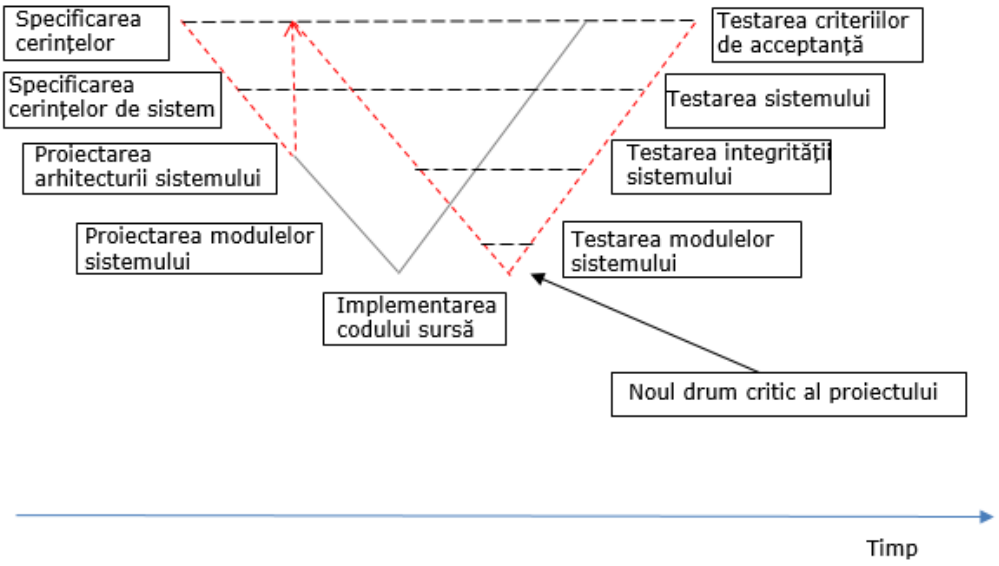

Fig. 4. 18 Drumul critic rezultat în urma modificării cerințelor.

Problemele generate de decalajul calendarului proiectului au condus autorul către găsirea soluțiilor în contracararea efectelor schimbării cerințelor în proiectele de dezvoltare a sistemelor de radio-navigație.

Identificând constrângerea sistemului ca fiind drumul critic al proiectului, trebuie găsită soluția elevării constrângerii prin diferitele instrumente avute în proiect. Cele mai la îndemâna instrumente folosite pentru elevarea constrângerii sunt:

- angajarea de noi experți în proiect;
- reducerea conținutului funcțiilor din proiect;

Din experiența autorului însă nici una din cele două soluții nu este viabilă.
Dezvoltarea produselor de înaltă calitate și a funcțiunilor noi sunt doi din principalii factori pentru ca proiectul să aibă succes.
Angajarea de noi experți va duce la efort suplimentar pentru restul componenților echipei datorită necesității integrării noilor membri în echipă.
Reducerea funcționalităților proiectului va duce la pierderea de clienți și automat a cotei de piață.

Trebuie urmărite astfel noi modalități de contracarare a efectelor schimbării cerințelor aplicând teoria constrângerii în proiecte.

Conform teoriei constrângerii trebuie găsite metode de elevare a constrângerii. În proiectele automotive software a sistemelor de radio-navigație nu există metode sau instrumente care să permită elevarea constrângerii. Aceasta a condus autorul la concluzia necesității de noi metodologii de lucru care elevează constrângerea.

În urma analizei aprofundate a literaturii de specialitate, din punctul de vedere al autorului elevarea constrângerii constă în dezvoltarea unui nou model de

implementare. Noul model de implementare combină metodele tradiționale de implementare cu metodologiile agile.

Analiza și implementarea noului model se realizează parcurgând pas cu pas modelul de dezvoltare în V și propunând metodologii agile care să înlocuiască respectiva fază. Modelul de dezvoltare astfel conturat va fi compus din însumarea tuturor soluțiilor găsite și implementate de-a lungul modelului de dezvoltare în V. Analiza metodologiilor agile și a efectului schimbării cerințelor asupra fiecărei faze a ciclului de viață în V folosit în dezvoltarea sistemelor de radio-navigație a condus la implementarea noului model de dezvoltare.

4.3.2.3 Analiza și implementarea modelului de dezvoltare AGILE la nivelul fazei de proiectare a arhitecturii sistemului de radio-navigație

Procesul schimbării cerințelor în proiectele automotive începe din faza de proiectare a arhitecturii [108]. Modificările cerințelor în această fază a unui proiect care utilizează ciclul de viață în V au următoarele consecințe:

- discuții între clientul și furnizorul proiectului legat de necesitatea implementării noii cerințe;
- riscul reproiectării parțiale sau totale a arhitecturii sistemului;

Motivul utilizării metodei Adaptive Software Development (ASD) în primele faze ale ciclului de viață în V își are originea în nevoia intensă de comunicare și clarificare a cerințelor la începutul proiectului. Aplicarea unei metodologii agile (ca de ex. ASD) care facilitează aplicarea modelului de colaborare client-furnizor, implica înțelegerea informațiilor, cu scopul bunei colaborări între parteneri [1-Badea, Proștean, Huțanu, Popa]. De altfel caracteristicile principale ale metodologiei agile ASD sunt:

- comunicarea în proiect;
- îndreptarea tuturor activităților către îndeplinirea obiectivelor proiectului;

Folosirea tehnicii Joint Application Development (JAD) în locul schimbului caietului de sarcini între clientul și furnizorul proiectului are ca rezultat descrierea funcțiilor sistemului precum și revizuirea funcțiilor sistemului de către client.

Proiectarea arhitecturii sistemului impune comunicare intensă între arhitecții modulelor proiectului, aceasta implică înlocuirea fazei de proiectare din cadrul ciclului de dezvoltare în V cu tehnica de colaborare a ASD.

Caracteristicile necesare și suficiente fazei de proiectare a arhitecturii sistemului sunt:

- dezvoltarea funcțională a sistemului software;
- toleranța la schimbări;
- reacție la risc;
- realizarea obiectivelor intermediare;

Dezavantajele utilizării metodelor clasice de proiectare a arhitecturii sunt:

- Discuțiile fără rezultat între diferiții arhitecți ai sistemului;
- Arhitecții sistemului aparțin unei echipe de dezvoltare, având rol dublu, vor încerca să proiecteze arhitectura astfel încât echipa de care aparține să aibă mai puțin de lucru;

- Neimplicarea clientului în proiectarea arhitecturii duce la neconfirmarea necesităților clientului. Aceasta va genera modificări ale cerințelor pe parcursul proiectului;
- Furnizorul proiectului nu primește confirmarea cerințelor din partea clientului proiectului;

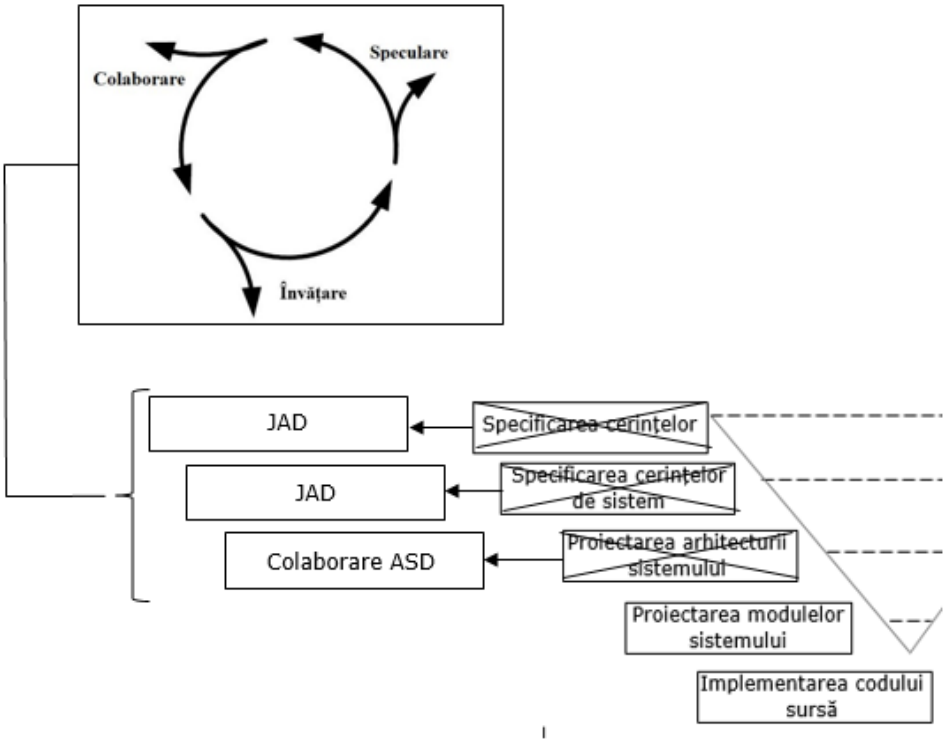

Fig. 4. 19 Metodologia ASD aplicată fazelor de specificare a cerințelor și de proiectare a arhitecturii sistemului

Concluzionând, principalul dezavantaj îl reprezintă lipsa comunicării cu scopul atingerii țelului fazei de dezvoltare. „Deseori clientul proiectului nu înțelege complexitatea sistemului în detaliu, dar prin informarea exactă asupra specificațiilor va contribui la prevenirea problemelor ulterioare"[76].

Avantajele înlocuirii primelor trei faze din ciclul de dezvoltare în V cu metodologia ASD și JAD sunt:

- Colaborarea intensă între clientul proiectului și echipa de dezvoltare într-o fază timpurie a proiectului;
- Capacitatea de predicție a influenței schimbării cerințelor asupra proiectului;
- Implementarea arhitecturii sistemului în așa fel încât să fie tolerantă la schimbări;
- Colaborare îmbunătățită între echipele proiectului datorită colaborării ASD;

Configurarea echipei si decizia asupra membrilor echipei de proiect este influențată de următoarele criterii [1 – Badea, Proștean, Hutanu, Popa]:

- Aptitudinea de rezolvarea a problemelor pe cale colaborativă;
- Coordonarea si prelucrarea informațiilor;
- Planificarea si Îmbunătățirea proceselor
- Coordonarea activităților

4.3.2.4 Analiza și implementarea modelului de dezvoltare AGILE la nivelul fazei de proiectare a modulelor sistemului de radio-navigație

Schimbarea cerințelor în faza de proiectare a modulelor sistemelor de radio-navigație va duce la întârzieri în proiect, întârzieri datorate necesității analizării și regândirii întregului sistem. Plecând de la noul model de dezvoltare prezentat în Fig. 4. 19, trebuie găsite soluții care să scurteze durata necesară derulării fazei de proiectare a modulelor sistemului.

Metodologia AGILE aleasă pentru a contracara eventualele întârzieri provocate de schimbarea cerințelor este Crystal Clear. Autorul a ales această metodologie deoarece în această fază fiecare modul este proiectat de echipe diferite. Aceasta face utilizarea metodologiei Crystal Clear foarte ușoară. Echipele de mici dimensiuni prezintă o flexibilitate mărită la schimbări precum și la transferul cunoașterii.

Alegerea metodologiei Crystal Clear se datorează în principal capacității metodologiei de a dezvolta proiecte incremental folosind echipe bine sudate. Acest avantaj aduce după sine alegerea în consens a priorităților fazei de proiectare. Deoarece metodologia Crystal Clear înlocuiește o singură fază a modelului în V, aceasta va fi adaptată necesității fazei. De exemplu în această fază se va elimina faza de testare a metodologiei. Fig. 4. 20 prezintă partea din stânga a modelului în V bazat pe Fig. 4. 19 rezultat în urma înlocuirii fazei și proiectare a modulelor a sistemului cu metodologia Crystal.

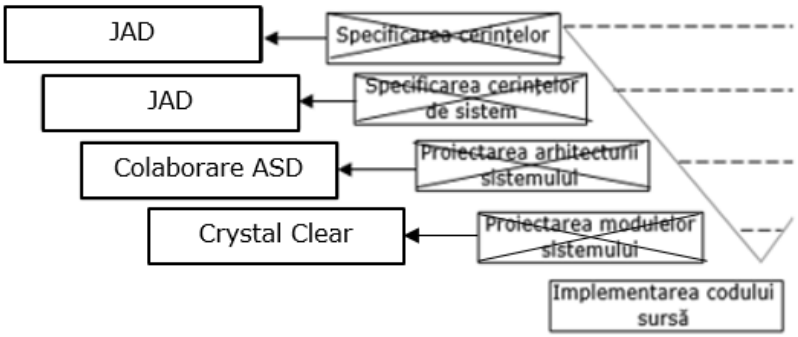

Fig. 4. 20 Metodologia Crystal aplicată fazei de proiectare a modulelor sistemului

Dezavantajele folosirii proceselor clasice de proiectare a modulelor sistemului sunt:

- Lipsa unui prototip, astfel încât să se poată elimina eventuale erori de proiectare;

- Neimplicarea clientului duce la neconfirmarea necesităților acestuia;
- Lipsa instrumentelor de măsurare a performanței fazei de proiectare a modulelor sistemului;

Tabel 4. 7 Caracteristici Crystal Clear folosite în noul model.

Caracteristici și tehnici agile	**Relevant Crystal Clear**	**Relevanță pentru această fază a noului model**
Livrarea incrementală a proiectelor	**X**	**X**
Urmărirea evoluției proiectului bazată pe livrările componentelor	**X**	**X**
Implicarea directă a clientului	**X**	**X**
Testarea automată a funcționalităților sistemului	**X**	
Organizarea de ședințe cu scopul de a îmbunătăți produsul și metodologia la începutul și mijlocul incrementului	**X** **X**	**X** **X**
Controlul versiunilor	**X**	**X**
Programare	**X**	
Comunicare	**X**	**X**
Măsurarea performanței	**X**	**X**

O comparație între caracteristicile Crystal Clear și caracteristicile preluate în noul model este prezentată în Tabel 4. 7.
Avantajele utilizării metodologiei Crystal în această fază sunt:

- livrările incrementale ale rezultatelor fazei în funcție de prioritățile proiectului;

- continuarea implicării clientului proiectului prin organizarea de întâlniri cu rolul de prezentare a rezultatelor intermediare;
- adaptabilitatea procesului. Prin organizarea de întâlniri cu rolul de analiză a rezultatelor intermediare se pot determina îmbunătățiri ale procesului;
- flexibilitatea la schimbări. Dimensiunile reduse ale echipelor de lucru facilitează transferul cunoașterii și comunicarea în proiecte;

4.3.2.5 Analiza și implementarea modelului de dezvoltare AGILE la nivelul fazei de implementare a codului sursă a sistemului de radio-navigație

Implementarea codului sursă presupune transcrierea cerințelor într-un limbaj de programare. Aceasta presupune ca arhitectura sistemului să fie definită, iar interfețele între modulele sistemului să fie cunoscute. Schimbarea cerințelor în această fază sau una din fazele premergătoare implementării codului sursă prezintă dezavantajul unor întârzieri însemnate proiectului datorită necesității redefinirii arhitecturii sau a interfețelor modulelor sistemului.

Modelul în V aplicat proiectelor sistemelor de radio-navigație prevede implementarea tuturor cerințelor după care se trece la rezolvarea erorilor. În funcție de cerințele care vor fi modificate, acestea atrag după sine în unele cazuri rescrierea întregului cod sursă. Alte dezavantaje ale fazei de implementare a codului sursă din cadrul modelului în V sunt:

- imposibilitatea de prioritizare a funcțiilor implementate pe termen scurt;
- în cazul ciclurilor de lungă durată, codul implementat va putea fi testat cu întârziere;
- în cazul modificării cerințelor există posibilitatea reimplementării a unei părți din codul deja implementat;
- inflexibilitate la schimbările cerințelor sau a priorităților;

Preîntâmpinarea acestor cazuri poate fi efectuată prin utilizarea metodologiei AGILE SCRUM în implementarea software. Alegerea acestei metodologii este determinată de flexibilitatea la schimbări demonstrată în alte proiecte.

Caracteristicile esențiale pentru a înlocui faza de implementare din modelul în V sunt:

- definirea “product backlog” precum și a “sprint backlog”;
- ședințele de planificare a „sprintului”. Aceste ședințe vor fi adaptate în funcție de necesitățile proiectului. Spre deosebire de procesul SCRUM clasic aceste ședințe nu sunt conduse de către un “product owner” ci de către arhitecții și responsabilii de integrarea sistemului. Motivul este dat de necesitatea de corelare a dependențelor modulelor;
- ședințele zilnice de monitorizare a stadiului implementării. Frecvența acestor ședințe va fi definită în funcție de mărimea sprintului și necesitatea proiectului;
- recenziile sprintului au rolul de a informa clientul proiectului asupra stadiului implementării;

Pornind de la Fig. 4. 20, Fig. 4. 21 prezintă grafic noul model de implementare în V utilizând metodologiile AGILE descrise anterior.

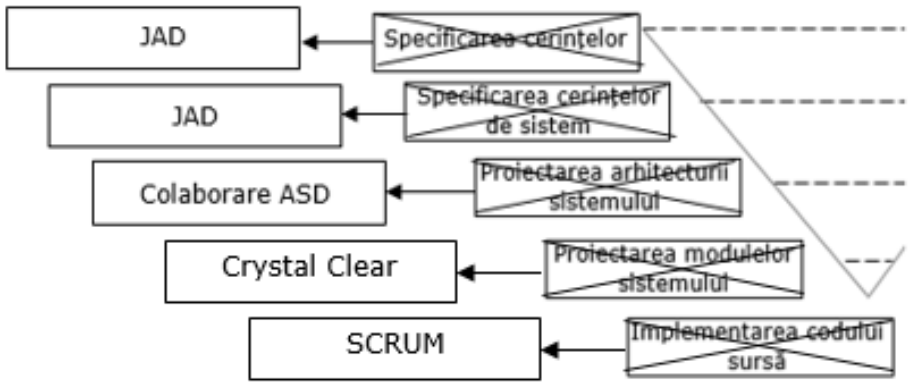

Fig. 4. 21 Metodologia SCRUM aplicată fazei de implementare a codului sursă

Avantajele utilizării metodologiei SCRUM în faza de implementare a codului sursă sunt:

- planificarea exactă a cerințelor ce urmează a fi implementate în următorul sprint;
- coordonarea implementării diferitelor module ale sistemelor de radio-navigație;
- monitorizarea stadiului implementării sistemului prin participarea clientului la ședințele de monitorizare a sprintului precum și la ședințele de revizuire ale sprintului;
- flexibilitate la prioritizare;

4.3.2.6 Analiza și implementarea modelului de dezvoltare AGILE la nivelul fazei de testare a modulelor sistemului de radio-navigație

Odată cu implementarea codului sursă se finalizează faza de construcție a modulelor sistemului. Testarea aferentă modulelor trebuie să corespundă metodologiei folosite de-a lungul liniei orizontale pe latura din stânga a modelului în V.

Testarea modulelor sistemului trebuie astfel să folosească strategii de testare din metodologiile AGILE Crystal Clear și SCRUM. Utilizarea metodologiilor AGILE de testare aduc cu sine avantajul testării individuale ale funcțiilor sistemului imediat după ce acestea au fost implementate. Rigiditatea modelul în V este datorat și neînceperii activităților unei faze dacă condițiile de intrare ale fazei premergătoare acesteia nu este realizată.

Testarea prin metodologiile AGILE permite identificarea erorilor de programare în stadii incipiente ale implementării și accelerarea implementării sistemului ca ansamblu. Testarea fiecărui caz de utilizare imediat după implementarea acestuia dă programatorilor șansa de a corecta „din mers” erorile apărute în urma fazei de testare a modulelor sistemului.

Incubarea fazei de testare în faza de implementare va crea modularitatea necesară implementării cerințelor noi apărute în timpul implementării caietului de

sarcini. Toate fazele anterior descrise sunt concepute în așa fel încât schimbările cerințelor să nu afecteze întreaga arhitectură a sistemului.

Avantajele utilizării metodologiilor AGILE în testarea modulelor sistemului sunt:

- Rezultate imediate ale testării cazurilor de utilizare. Crearea rezultatelor se datorează testării cazurilor de utilizare imediat după implementarea acestora;
- Modularitatea dată de strategia de testare menționată anterior;
- Reacția la modificări și schimbări ale priorităților;
- Posibilitatea de raportare a stării proiectului în orice moment;

Fig. 4. 22 prezintă modelul de dezvoltare până la faza de testare a modulelor sistemului, folosind metodologiile AGILE în locul fazelor tradiționale utilizate în modelul de implementare în V.

Între faza de implementare a codului sursă și faza de testare a modulelor sistemului există o relație de sincronizare. Raportarea și transmiterea rezultatelor testării prin diferite instrumente (Fig. 4. 22 1. și 2.) concretizează căile de comunicare între echipa de programatori și echipa de testare, eficiența comunicării fiind esențială în transmiterea informațiilor către echipa de programatori.

Schimbarea cerințelor în faza de testare a modulelor aduce după sine modificări ale fazelor anterior definite în proiect. Datorită incubării fazei de testare în faza de implementare a codului sursă și în funcție de stadiul implementării, schimbarea cerințelor poate avea urmări diferite asupra proiectului. Analiza impactului schimbării cerințelor asupra proiectului se va efectua începând din faza de JAD a modelului.

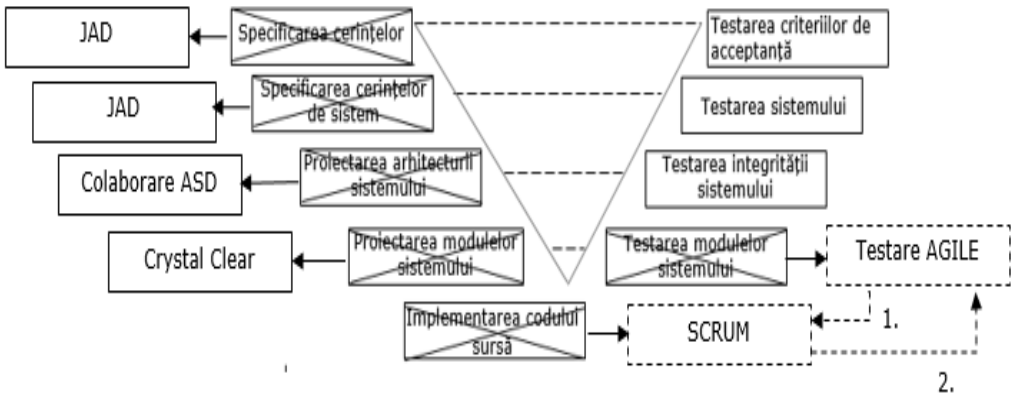

Fig. 4. 22 Metodologia SCRUM aplicată fazei de testare a modulelor sistemului

4.3.2.7 Analiza și implementarea modelului de dezvoltare AGILE la nivelul fazelor de testare a integrității sistemului, testării sistemului sau a criteriilor de acceptare

Fazele testării integrității sistemului, testării sistemului sau a testării criteriilor de acceptare presupune existența sistemului ca ansamblu. Schimbarea cerințelor într-o fază atât de târzie presupune analiza din prima fază a modelului de dezvoltare, indiferent dacă acesta este un model tradițional sau un model care folosește metodologii AGILE.

Necesitatea analizei și modificării sistemului atrage după sine obligația de retestare a întregului sistem. Aceste precondiții au condus autorul către concluzia că fazele de testare ale modelelor de dezvoltare tradiționale nu trebuie înlocuite cu metodologii AGILE de testare, argumentând în special cu faptul ca modelul AGILE corespondent pe latura stângă nu este decât unul de îmbunătățire a comunicării și nu a tehnologiei aplicate în ingineria programării.

Analiza incrementală a modelului de dezvoltare în V și înlocuirea fazelor cu o metodologie AGILE care se potrivește sistemelor de radio-navigație automotive, face ca Fig. 4. 22 să reprezinte și modelul de dezvoltare AGILE final aplicat modelului de dezvoltare în V.

Ciclurile AGILE JAD, JAD, Crystal Clear, SCRUM (2JCS) inserate și prezentate în cadrul acestui capitol, care înlocuiesc fazele celor două laturi ale ciclului de viață clasic în „V" reprezintă modelul original conceput de către autor în cadrul acestei teze, care vin în sprijinul gestiunii proiectelor software de radio-navigație supuse variațiilor generate de toate strategiile de viteză prezentate anterior.

4.4 Concluzii

Alegerea modelului de dezvoltare a unui proiect poate fi decisiv în succesul proiectului. Capitolul de față conține o analiză aprofundată a efectelor modificării cerințelor asupra calendarului proiectului precum și definirea măsurilor care să contracareze aceste efecte.

Implementarea noului model de dezvoltare tolerant la schimbări ale cerințelor are la bază următoarele analize:

- modificarea sau schimbarea cerințelor, în orice fază a ciclului de dezvoltare a proiectului reprezintă deja o certitudine în proiectele de dezvoltare software din industria automotive;
- utilizarea metodelor de dezvoltare tradiționale în producție, metode care stau și la baza dezvoltării proiectelor software automotive au la bază modele, care nu permit modificarea semnificativă a cerințelor de-a lungul dezvoltării proiectelor;
- sinteza dezavantajelor utilizării modelelor tradiționale în proiectele software din domeniul sistemelor de radio-navigație;
- metodele moderne ca de ex. „Teoria Constrângerilor" dezvoltată de E.M. Goldratt oferă explicații asupra neajunsurilor proceselor tradiționale în dezvoltarea software;
- explicația asupra neajunsurilor proceselor tradiționale în dezvoltarea software poate fi nuanțată prin sinteza factorilor de risc și decizionalcare duc la implementarea unei cerințe noi sau modificate cu scopul

propunerii unei soluții astfel încât calendarul proiectului să nu fie influențat;

Contribuțiile personale ale autorului sunt:

- analiza și descrierea concluziilor literaturii de specialitate legate de stabilitatea cerințelor în proiectele software;
- conceperea unei metode de decizie bazată pe formula matematica asupra oportunității implementării cerințelor noi sau modificate în dezvoltarea proiectelor de tip automotive;
- analiza proceselor de dezvoltare automotive și influența proceselor de producție asupra acestora;
- analiza și descrierea teoriei constrângerilor cu scopul validării noului model de dezvoltare;
- analiza și elaborarea factorilor care determină implementarea cerințelor noi;
- analiza și elaborarea factorilor care determină implementarea cerințelor noi;
- **configurarea unui nou model de dezvoltare, flexibil la schimbări, cu scopul de atenuare a efectului schimbării cerințelor în dezvoltarea proiectelor de tip automotive (modelul 2JCS).**

5. VALIDAREA MODELULUI DE DEZVOLTARE 2JCS

Obiectivul capitolului de față urmărește două direcții de validare a modelului 2JCS. Prima direcție evidențiază performanța modelului de dezvoltare 2JCS din punctul de vedere al încadrării proiectelor de radio-navigație în eșalonarea calendaristică inițială, indiferent de variațiile specificațiilor inițiale. A doua direcție de validare prezintă o metodă proprie de evidențiere a eficienței economice, prin prisma combinației celor două variabile economice „cost" și „timp" determinată de aplicarea modelului 2JCS. Capitolul a fost structurat în patru părți, după cum urmează.

Prima parte prezintă criteriile pe care trebuie să le îndeplinească un model de dezvoltare software și care au stat la baza validării modelului.

Partea a două exprimă metoda de validare concepută și motivează alegerea făcută.

Partea a treia realizează o analiză comparativă a două companii, care utilizează modele de dezvoltare diferite și evidențiază performanța timpului celor două proiecte.

Ultima parte a acestui capitol evidențiază performanța cele două modele de dezvoltare din punct de vedere financiar.

5.1 Considerații generale în validarea modelelor de dezvoltare

Implementarea modelelor de dezvoltare pentru proiectele software presupune găsirea soluțiilor optime pentru validarea unui sistem într-un mediu dedicat. „Validarea modelelor este importantă pentru dezvoltarea continuă a teoriei modelării și implementarea de noi modele precum și pentru practica inginerească cu țelul de a defini care metodologii și când să fie aplicate" (Els Du Bois)[31]. Din literatura de specialitate reiese de cele mai multe ori că deși validarea metodelor inginerești presupune aplicarea modelelor matematice, există însă situații de modele de dezvoltare inginerești în care aplicarea modelelor matematice nu vor duce la validarea noilor metodologii; „multe cercetări sunt bazate pe modele matematice, acest tip de validare au funcționat – și încă funcționează – foarte bine. Există însă domenii ale cercetării inginerești care se bazează pe afirmații subiective precum și pe modele matematice, ceea ce face ca validarea formală, riguroasă și cantitativă sa fie problematică" (Pederson et al. 2000) [100].

Obiectivul principal al modelelor de dezvoltare îl reprezintă gestiunea proiectelor software de radio-navigație, astfel încât să se economisească timp și costuri. Validarea modelelor noi este însă dificilă fără a avea o validare empirică a acestora. Acesta este motivul pentru care modelul nou creat trebuie să se dovedească eficient din punctul de vedere al combinației variabilelor economice „cost" și „timp" pe tot parcursul evoluției proiectului. De altfel, fiecare model trebuie să fie dezvoltat cu un anumit scop (Sargent 1984) [105] [31], iar validarea cercetării este un proces de construcție a încrederii în utilitatea acesteia respectând

scopul acesteia (Seepersad et al. 2005) [106]. Deoarece fiecare model este implementat conform nevoilor unui anumit domeniu, „validarea depinde de scopul metodologiei precum și de domeniul în care va fi aplicat" (Macal, 2005) [88] [31]. Problemele principale în validarea modelelor conform Kitchenham (1997) [75] sunt:

- Imposibilitatea definirii standardelor de control necesare;
- Probleme de scalare;
- Costuri de evaluare;
- Probleme în a defini beneficiile pentru anumite tipuri de tehnologie;
- Măsuri imature;
- Dificultăți de eliminare a factorilor care duc la neînțelegerea problemei;

Pentru ca un model să poată fi validat, acesta trebuie conform literaturii de specialitate să îndeplinească mai multe criterii:

- Funcționalitate: scopul modelului trebuie să fie cunoscut și să fie îndeplinit;
- Credibilitate(Eisner 1991)[30][31]: presupune îndeplinirea criteriilor importante fiecărui domeniu astfel încât încrederea aplicării modelului să fie mare;
- Logic (Sargent 1984) [105], (Olewnik și Lewis 2003)[96][31]: rezultatele obținute în urma folosirii modelului să aibă sens;
- Complet (Olewnik și Lewis 2003) [96][31];
- Inteligibil (Olewnik și Lewis 2003) [96][31];
- Folositor (Olewnik și Lewis 2003) [96][31];
- Consistent (Olewnik și Lewis 2003) [96][31];
- Empiric (Olewnik și Lewis 2003) [96][31];
- Corespundă specificațiilor (Olewnik și Lewis 2003) [96][31];
- Aducă valoare adăugată (Olewnik și Lewis 2003) [96][31];
- Validitate descriptivă (Maxwell 1992): descriere validă a modelului fără a distorsiona adevărul [91][31];
- Validitate interpretativă (Maxwell 1992): interpretarea din punct de vedere valoric a modelului [91][31];
- Validitate teoretică (Maxwell 1992): reprezintă validitatea conceptuală a modelului [91][31];
- Generalizabil (Maxwell 1992): reprezintă aplicabilitatea modelului de dezvoltare asupra altor domenii [91][31];
- Validitate evaluativă (Maxwell 1992): reprezintă evaluarea validității unui model de dezvoltare [91][31];

Metodele de validare ale unui model de dezvoltare se diferențiază în funcție de domeniul în care trebuie să funcționeze acesta. Conform literaturii de specialitate există mai multe metode /strategii de validare a unui mode:

- Recenzie (Kitchenham 1997): reprezintă evaluarea detaliată a modelului de către un expert [75];
- Studiu de caz simulat (Kitchenham 1997): reprezintă rularea sau simularea modelului conform planului de validare [75];

- Studiu de caz real Kitchenham(1997): reprezintă aplicarea modelului de dezvoltare asupra unui proiect real [75];
- Rularea de sondaje (Gottschalk 2002): presupune evaluarea conform rezultatelor obținute în urma completării sondajelor asupra modelului de dezvoltare [39][31];
- Comparație cu alte modele (Sargent 1984 [105], Carson II 2002 [67]): presupune rularea modelelor în aceleași condiții de lucru și compararea rezultatelor acestora;
- Teste degenerative (Sargent 1984 [105]): presupune introducerea de valori neașteptate sau îndepărtarea anumitor faze a modelului pentru a demonstra necesitatea fazelor respective;
- Validarea predictivă (Sargent 1984 [105]): se utilizează modelul nou dezvoltat cu scopul de a prezice comportamentul sistemului (în cazul nostru al proiectului) în viitor;

5.2 Alegerea metodei de validare.

Analiza metodelor de validare a modelelor de dezvoltare software din literatura de specialitate a condus la concluzia că nu există o metodă dedicată care să poată fi folosită direct. De altfel Els Du Bois [31] declara că „încă nu există metodologii efective care să valideze modelul în contextul nostru". Prin „modelul nostru" Els du Bois [31] nu se referea la un model de dezvoltare software aplicat proiectelor automotive, ci se referea la un model de dezvoltare specific lucrării sale. De altfel și Kitchenham (1997) [75] spune că nu există o metodă general valabilă care să fie întotdeauna cea mai bună.

Prin obiectivul acestui paragraf de a valida modelul propus 2JCS **J**AD – **J**AD - **C**olaborare ASD-**C**rystal Clear – **S**CRUM- Testare **S**CRUM), autorul a cercetat în literatura de specialitate metode posibile de validare a modelului. Unul dintre cele mai utilizate modele de validare este soluția „black-box" și/sau „white box". Deoarece rezultatele găsite nu au putut fi aplicate asupra modelului de dezvoltare propus (2JCS), autorul și-a îndreptat atenția către găsirea unei companii care să preia și să implementeze modelul 2JCS în proiectele sale. S-a realizat astfel posibilitatea de comparație între performanța modelului clasic de dezvoltare în V aplicat în compania „V" și performanța modelului 2JCS aplicat în compania „A". Din motive de confidențialitate companiile în care s-a experimentat în mod comparativ performanta celor două modele au fost numite „V" (de la modelul de dezvoltare în V) respectiv „A" (de la metodologiile AGILE aplicate modelului). Pe lângă validarea performantei modelului 2JCS din punctul de vedere al încadrării proiectelor în timp se realizează validarea pe o noua axă, și anume validarea din punct de vedere financiar. Cele două proiecte cu metodologii diferite de implementare au avut dimensiuni apropiate. Numărul cerințelor în proiectul din cadrul companiei V a fost de aproximativ 11000 iar în cadrul companiei A de aproximativ 11700.

Metoda de validare presupune compararea fiecărei faze a proiectelor, un proiect utilizând modelul de dezvoltare în V iar celălalt modelul de dezvoltare 2JCS. În final se compară cele două modele din perspectiva parcurgerii, respectiv, trecerii dintr-o etapă în alta. Comparația modelelor se realizează doar pentru acele faze ale modelului în V care au fost înlocuite cu metodologii AGILE.

Validarea din punct de vedere financiar se realizează prin intermediul analizei pierderilor provocate de întârzierile acumulate în proiectele automotive, care utilizează modelul de dezvoltare în V.

5.3 Validarea modelului 2JCS din punct de vedere al încadrării proiectelor de radio-navigație în eșalonarea calendaristică inițială

Validarea empirică a modelului propus din punct de vedere temporal se realizează prin analiza comparativă de faze corespunzătoare modelului în V, respectiv, modelului 2JCS. Parcursul validării cuprinde atât analize comparative între cele două modele pentru faze singulare, cât și pentru faze combinate.

5.3.1 Analiza comparativă între modelul în V și modelul 2JCS pentru faze de creare a cerințelor proiectului și a cerințelor sistemului

Activitățile de realizare a caietului de sarcini precum și introducerea acestuia în sistemul intern al furnizorului reprezintă în fapt o activitate „înfrățită", ceea ce necesită privirea celor două acțiuni ale ciclului în V ca pe o singură fază (fig. 5.1). Prin comparație, schimbul de informații referitoare la cerințele inițiale ale proiectului, precum și cele legate de introducerea acestora în sistem sunt de asemenea privite ca o singură fază în metodologia JAD, utilizată în modelul 2JCS. JAD determină organizarea de întâlniri periodice și dese astfel încât cerințele să fie sincronizate direct intre clientul și furnizorul sistemului. Datele colectate sunt rezultatul experienței practice a autorului cat și a discuțiilor purtate de acesta cu diferiți coordonatori de proiect și programatori care au aplicat modelul descris mai sus.

În proiectul din cadrul companiei V în care s-a folosit modelul de dezvoltare în V, timpul necesar realizării și coordonării cerințelor a fost de 51 de săptămâni. Schimbul de informații s-a realizat pe canale oficiale cu ajutorul instrumentelor de schimb de date. Totodată nu s-au realizat ședințe cu scopul de sincronizare a informațiilor intre clientul și furnizorul proiectului.

Activitățile realizate în aceasta faza au fost:

- Crearea specificațiilor inițiale: **24 săptămâni**;
- Revizuirea specificațiilor: **5 săptămâni**;
- Alegerea cerințelor relevante pentru proiect: **6 săptămâni**;
- Predarea cerințelor către furnizor: **1 săptămână**;

Faza creării cerințelor de sistem în cadrul aceluiași proiect al companiei V a fost de 16 săptămâni și a fost compusa din următoarele faze:

- Analiza cerințelor: **6 săptămâni**;
- Revizuirea cerințelor: **2 săptămâni**;
- Cerere modificare cerințe: **1 săptămâna**;
- Revizuire cerere modificare cerințe: **1 săptămâna**;
- Modificare cerințe: **2 săptămâni**;
- Revizuire cerințe modificate: **1 săptămâna**;
- Livrare cerințe modificate: **1 săptămâna**;

- Livrare cerințe către arhitecții sistemului: **1 săptămâna**;

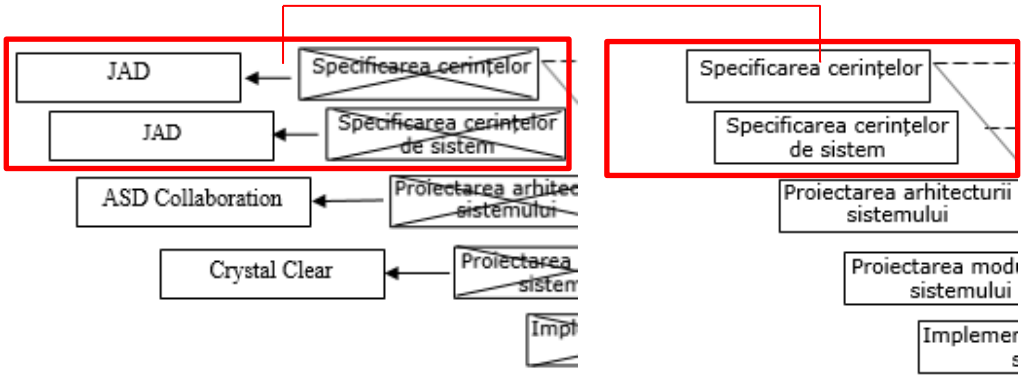

Fig. 5. 1 Comparație între fazele de specificare a cerințelor a modelelor 2JCS și V.

Pe lângă timpul necesar efectuării activităților descrise mai sus, proiectul a necesitat timp suplimentar, fiind afectat de întârzierilor provocate de proiecte derulate în paralel. Concret, între faza de „revizuire a cererii de modificare a cerințelor" (1 săptămână) și a fazei „modificării cerințelor"(2 săptămâni) s-au pierdut încă **2 săptămâni** din pricina capacității reduse a echipei, respectiv din nevoia de rezolvare a unor activități cu prioritate mai mare. În cadrul acestui exemplu cele două faze s-au derulat pe parcursul a 5 săptămâni în loc de 3 săptămâni conform estimării inițiale a proiectului, prezentate mai sus.

În comparație cu proiectul V descris mai sus, proiectul A, care a utilizat metodologia AGILE JAD a avut următoarele subfaze:

- Crearea specificațiilor inițiale: **24 săptămâni** (eșalonarea calendaristică a fost tot de 24 de săptămâni, faza fiind preluată din modelul în V, compania A și-a dorit ca specificațiile inițiale să nu fie externalizate);
- Revizuirea specificațiilor: **5 săptămâni** (preluare din modelul în V -> compania A și-a dorit ca specificațiile inițiale să nu fie externalizate);
- Alegerea cerințelor relevante pentru proiect: **6 săptămâni** (preluare din modelul în V -> compania A și-a dorit ca specificațiile inițiale să nu fie externalizate);
- Predarea cerințelor către furnizor s-a realizat în ședința de sincronizare între client și furnizor din cadrul metodologiei JAD. În cadrul aceleiași ședințe s-au parcurs cerințele definite în pasul anterior cu scopul realizării analizei inițiale a cerințelor. Durata timpului care a fost necesara acestei faze a fost de **3 zile**.
- Analiza detaliată a cerințelor s-a realizat de către furnizor în urma ședinței din pasul anterior, această fază fiind contopită cu faza de revizuire a cerințelor. Timpul parcurs pentru această fază, de către furnizor a fost de **2 săptămâni**;

- Următorul pas abordat în proiect a fost organizarea unei noi întâlniri în care au fost discutate cererile de revizuire a cerințelor. Modificările dorite de către furnizor asupra cerințelor s-au efectuat rapid în cadrul **unei săptămâni**. Întâlnirile de sincronizare au dus la înțelegerea limitărilor arhitecturale ale proiectului, ceea ce a coincis cu acceptul cererii de modificare a cerințelor. Timpul cumulat parcurgerii acestei faze a fost necesar din cauza nevoii de actualizare a documentației din instrumentele de stocare a cerințelor;
- Într-o a treia întâlnire s-au discutat ultimele detalii legate de specificațiile de sistem, cum ar fi:
  - modificările aduse cerințelor în urma cererii înmânate de către furnizor
  - livrarea oficială a acestor cerințe
  - înmânarea documentației cerințelor către arhitecții sistemului (3 zile).

Pe lângă timpul necesar parcurgerii fiecărei subfaze, s-au identificat durate de timp pierdute, datorate pregătirii întâlnirilor de sincronizare între client și furnizor. Timpul total pierdut în urma pregătirii ședințelor de sincronizare a fost de **3 săptămâni**. Aceste întârzieri au fost cauzate în principal de lipsa viziunii asupra duratei necesare prelucrării informațiilor noi apărute în cadrul ședințelor de sincronizare.

În tabelul 5.1 se realizează comparația duratelor necesare fazelor celor doua modele analizate.

Tabel 5. 1 Comparație a 2 proiecte automotive utilizând modele de dezvoltare diferite în faza de specificare a cerințelor

.Activități ale fazei de definire a cerințelor	**Timp necesar în cadrul companiei V (în săptămâni)**	**Timp necesar în cadrul companiei A (în săptămâni)**
Creare Specificații inițiale	24	24
Revizuire specificații	5	5
Alegerea cerințelor relevante pentru proiect	6	6
Predarea cerințelor către furnizor	1	0,5
Analiza cerințelor	6	1
Revizuirea cerințelor	2	1
Cerere modificare cerințe	1	1
Revizuire cerere modificare	1	0
Modificare cerințe	2	0
Revizuire cerințe modificate	1	0

Livrarea cerințe modificate	1	0.5
Livrare documente cerințe către arhitecții sistemului	1	0
Timp utilizat în parcurgerea fazelor	51	39
Timpi tranziție intre activități	2	3
Timp necesar parcurgerii fazei specificării cerințelor (Suma subfazelor)	**53**	**42**

În concluzie, timpul total necesar parcurgerii celor două faze în cadrul proiectului în compania V este de 53 săptămâni, cu **11 săptămâni (20,7%)** mai mult decât în cazul utilizării modelului 2JCS din cadrul companiei A (Tabel 5.1).

În sprijinul concluziei de mai sus câțiva autori ai literaturii de specialitate își prezintă rezultatele altor proiecte similare și experiențe practice, care exprimă o economie de timp între 15% [60] și 75%-90% [55], [58]. Diferența mare obținută între rezultatul practic (20,7%) (comparația între modelul în V și 2JCS) și cel teoretic (75%-90%)[55][58] este cauzată de prudența companiei A de a nu furniza informații complete asupra specificațiilor încă din prima fază a proiectului. Cu scopul protejării anumitor informații, specificațiile au fost create asemănător cu ciclul tradițional de dezvoltare, metodologia JAD fiind aplicata abia după derularea sincronizării interne companiei V.

5.3.2 Analiza comparativă între modelul în V și modelul 2JCS pentru faze de proiectare a arhitecturii sistemului

Asemenea validării performanței modelului 2JCS în faza de specificare a cerințelor și a creării cerințelor de sistem, faza de proiectare a arhitecturii a fost de asemenea validată prin compararea timpilor necesari rulării acestei faze în cadrul companiei V și a companiei A (fig. 5.2). Faza de proiectare a arhitecturii modelului în V este total diferita față de cea din modelul 2JCS. În cadrul modelului în V, faza de proiectare a arhitecturii sistemului este o fază dedicată, plasată în cadrul secvenței din latura din stânga. În cadrul modelului 2JCS această fază de proiectare a arhitecturii se realizează pe tot parcursul ciclului modelului, acceptând o actualizare continuă datorată schimbărilor cerințelor proiectului.

În cazul proiectului din cadrul companiei V, faza de proiectare a arhitecturii a fost împărțită în următoarele 3 subfaze principale:

- Analizarea cerințelor de sistem: **3 săptămâni**;
- Crearea cerințelor aritecturale:**12 săptămâni**;
- Revizuirea cerințelor arhitecturale și a arhitecturii sistemului: **3 săptămâni;**

Analiza comparativă între cele două modele s-a realizat pe baza analizei informațiilor obținute de la membrii echipei de arhitecți din ambele companii. Conform arhitecților companiei A, utilizarea colaborării ASD cu scopul definirii

arhitecturii sistemului nu a condus la diminuarea timpului de rulare a fazei actuale, în schimb s-a obținut flexibilitatea necesară implementării cerințelor noi sau modificate din timpul proiectului

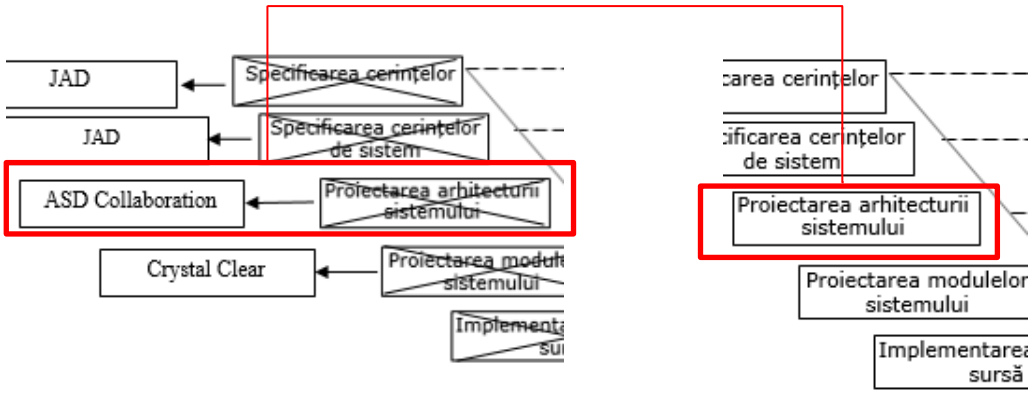

Fig. 5. 2 Comparație între fazele de proiectare a arhitecturii sistemului a modelelor 2JCS și V.

De asemenea, în teoria metodologiilor AGILE, definirea arhitecturii unui sistem este dificilă. În urma discuției autorului cu arhitecții sistemului companiei A, aceștia au declarat că principalul avantaj este acela de a împărți faza proiectării arhitecturii pe toata durata proiectului, în funcție de prioritățile acestuia. Astfel a fost posibilă implementarea funcțiilor de bază a sistemului încă din faza incipientă a proiectului, urmând ca pe parcursul dezvoltării ulterioare a acestuia, arhitectura concepută în mod flexibil să permită obținerea atât a cerințelor inițiale cât și a celor modificate pe parcursul proiectului.

5.3.3 Analiza comparativă între modelul în V și modelul 2JCS pentru faza de proiectare a modulelor software a sistemului

Durata fazei de proiectare a modulelor sistemului este relativ scurtă comparativ cu durata totală a proiectului (fig. 5.3). În cazul utilizării modelelor clasice de implementare în cadrul companiei V, durata acestei faze a fost de 3 săptămâni. Principala activitate a fost aceea de specificare a interfețelor interne modulelor sistemului. Dezavantajul major al acestei metode a fost lipsa de comunicare între membrii echipei, ceea ce a dus deseori la greșeli de proiectare și care au generat automat modificări, care nu erau planificate inițial.

Folosirea metodologiei AGILE Crystal Clear în cadrul fazei de proiectare a modulelor sistemului a remediat problema comunicării în cadrul echipei. Folosirea acestei metodologii a sporit comunicarea în cadrul echipei precum și eficiența review-ului. Arhitecții modulelor sistemului declarau că timpul necesar rulării acestei faze a fost apropiat din punct de vedere temporal cu cea a proiectelor care utilizau modelul în V, utilizarea metodologiei Crystal Clear a economisit din timpul proiectului (chiar daca necuantificabil) doar prin prisma reducerii erorilor de proiectare.

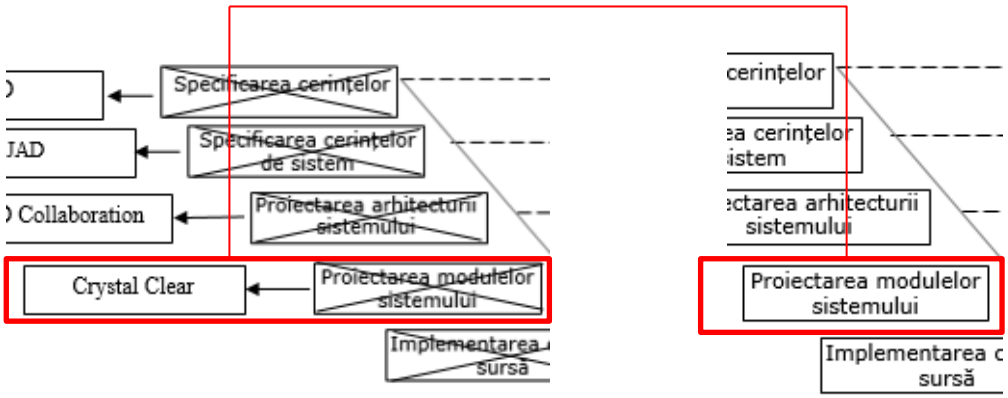

Fig. 5. 3 Comparație între fazele de proiectare a modulelor sistemului a modelelor 2JCS și V.

5.3.4 Analiza comparativă între modelul în V și modelul 2JCS pentru faza de implementare a codului sursă

Timpul necesar implementării codului sursă în faza de construcție a sistemului, utilizând modele tradiționale, este de un an, iar faza de rezolvare a erorilor este de 6 luni în cazul proiectelor automotive de mari dimensiuni. Acești timpi au rezultat în urma planificării tuturor cerințelor sistemului precum și din necesitatea de livrare a sistemului, astfel încât să corespundă calendarului de dezvoltare a mașinii.

Organizarea proiectului a fost de tip „Top-Down”, prioritățile de implementare au fost impuse de către coordonatorul de proiect în urma promisiunilor făcute clientului proiectului. Ordinea de implementare a cerințelor nu a respectat o regulă strictă, ci a fost creată astfel încât să corespundă țelurilor intermediare ale proiectului. De exemplu, într-o primă prezentare a produsului, coordonatorul de proiect a promis implementarea a 10% din cerințe, fapt ce a dus la implementarea cerințelor într-o ordine aleatorie, cu scopul de a satisface promisiunea făcută. Ordinea de implementare a funcționalităților nu a fost dată de perspectiva dezvoltării coerente a produsului. Astfel primele cerințe implementate erau cele „ușoare”, în locul celor „stabile”, cu potențial redus de modificare de-a lungul proiectului. Această strategie a dezvoltării proiectelor a dus la imposibilitatea acceptării la cererile de schimbare a cerințelor din partea clientului.

Pentru atingerea obiectivelor temporale, compania V a fost nevoită deseori să respingă cererile de schimbare a cerințelor. Negocierile intre client și furnizor au dus des la reducerea funcționalității sau acceptarea erorilor sistemului automotive în schimbul implementării unor cerințe modificate.

Aplicarea metodologiei AGILE SCRUM a adus avantajul flexibilității la schimbări prin aplicarea predictibilității cerințelor în funcție de gradul de risc al modificării acestora.

În urma discuțiilor cu programatorii implicați în dezvoltarea proiectului cu ajutorul metodologiei SCRUM, aceștia au confirmat productivitatea crescută în comparație cu proiectele la care au participat și care utilizau modelul în V. Totodată

aceştia au lucrat cu motivaţie crescută datorită capacităţii metodologiei de a transfera cunoştinţele în cadrul echipei.

Din punct de vedere temporal, utilizarea metodologiei SCRUM a redus timpul de implementare doar prin prisma implementării cerinţelor cu prioritate mare şi reducerea software-ului „rebut". Spre deosebire de proiectele care utilizau modelul în V, priorităţile proiectului au fost decise împreună cu clientul proiectului (fig. 5.4).

Programatorii companiei A, care au participat în trecut şi la proiecte utilizând modelul în V au declarat că diferenţa majoră între cele două modele o reprezintă modul de organizare a ordinii de implementare a cerinţelor. În cadrul companiei A, prioritatea implementării şi integrării funcţiilor era dată de probabilitatea maximă a stabilităţii cerinţelor. Primele funcţii implementate au fost acelea cu riscul cel mai mic de a fi modificate ulterior de către client. Aceeaşi programatori dădeau ca exemplu că primele module la care au lucrat au fost acelea de „Audio", a protocoalelor de comunicare cu diferitele componente ale maşinii care erau deja în serie, precum şi la submodule a căror utilizare pe piaţă devenise un standard (de exemplu Radio FM). Fig. 5.5 prezintă modul în care s-a ales prioritatea funcţiilor sistemului.

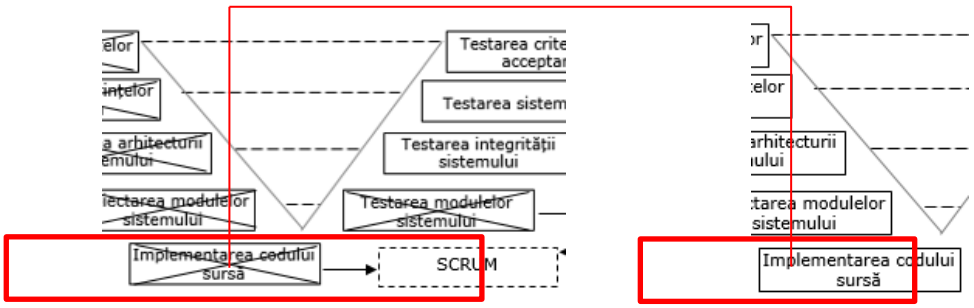

Fig. 5. 4 Comparaţie între fazele de implementare a codului sursă a modelelor 2JCS şi V.

Dacă în cadrul companiei V, modificările cerinţelor aduceau după sine întârzieri proiectului (Fig. 3.22), alegerea metodologiei SCRUM a dus la reorganizarea priorităţilor proiectului. Conform programatorilor companiei A, aplicarea metodologiilor SCRUM nu a dus la economisirea de timp, ci această metodologie a redus substanţial implementarea de cod „rebut" precum şi o prioritizare a funcţiilor încă din fazele incipiente ale proiectului, ceea ce a dus la o productivitate crescută.

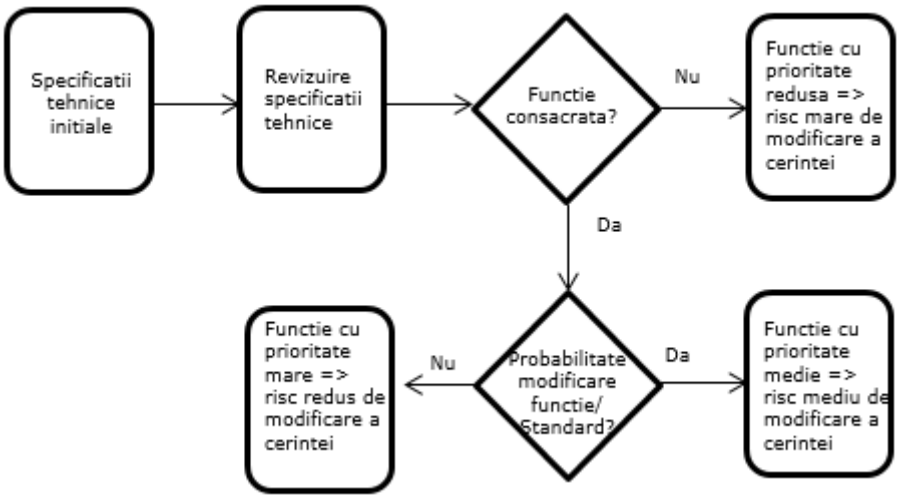

Fig. 5. 5 Modalitatea de alegere a priorităților funcțiilor în cadrul companiei A

Tot în urma discuțiilor cu programatorii și coordonatorii de proiect din cadrul companiei A a rezultat fig. 5.6. Aceasta prezintă grafic efectul modificării cerințelor asupra stivei priorităților cerințelor proiectului în cadrul companiei A. Orice modificare de cerință a fost considerată ca având prioritate mare și a fost planificată pentru următorul sprint. Pachetele de modificări au fost notate în fig. 5.6 cu „*", „+" și „#", prioritatea acestora putând fi observată în stiva priorităților. Introducerea stivei de priorități în consens cu cerințele clientului a dus la economisirea de timp. În estimarea programatorilor companiei A, economia de timp a fost de aproximativ 20% față de proiectele în care au utilizat modele tradiționale de dezvoltare. Aceștia explicau aceasta îmbunătățire prin faptul că o activitate începută nu era întreruptă, cerințele implementate erau în concordanță cu prioritățile clientului precum și atmosfera mai bună din cadrul proiectului.

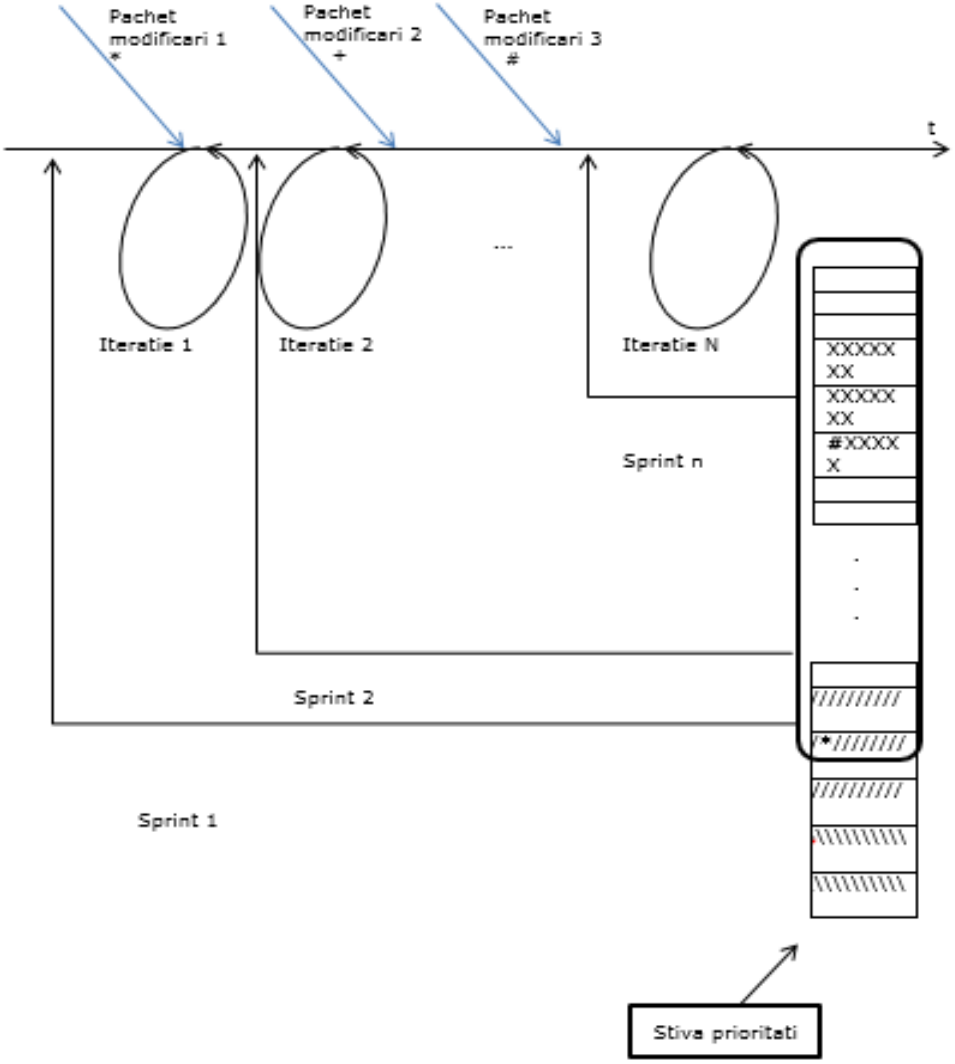

Fig. 5. 6 Efectul modificării cerințelor asupra desfășurării proiectului în cadrul companiei A

Aplicarea 2JCS în faza de implementare a codului sursă nu aduce cu sine o scurtare a timpului de desfășurare a proiectului, ci favorizează implementarea mai multor cerințe stabile, la care se adaugă cele modificate din cauza schimbărilor tehnologice.

Pe lângă rezultatele practice strânse de la compania A, care evidențiau avantajul utilizării modelului 2JCS, autorul a reușit să valideze partea de implementare a codului sursă și prin analiza literaturii de specialitate. Astfel Benediktsson et. al [13] a ajuns la concluzia că numărul liniilor de cod implementate aplicând modelul în V este mult mai redus decât dacă s-ar utiliza o metodologie agile.

5.3.5 Analiza comparativă între modelul în V și modelul 2JCS pentru faza de testare a modulelor sistemului

Analiza comparativă pentru faza de testare a modulelor (fig. 5.7) sistemului aplicate prin modelele utilizate în compania V și compania A nu a adus diferențe semnificative. Dacă în cadrul companiei V, testarea modulelor s-a realizat în urma implementării complete a unui modul, în cadrul companiei A testarea s-a realizat imediat ce codul sursă a fost implementat.

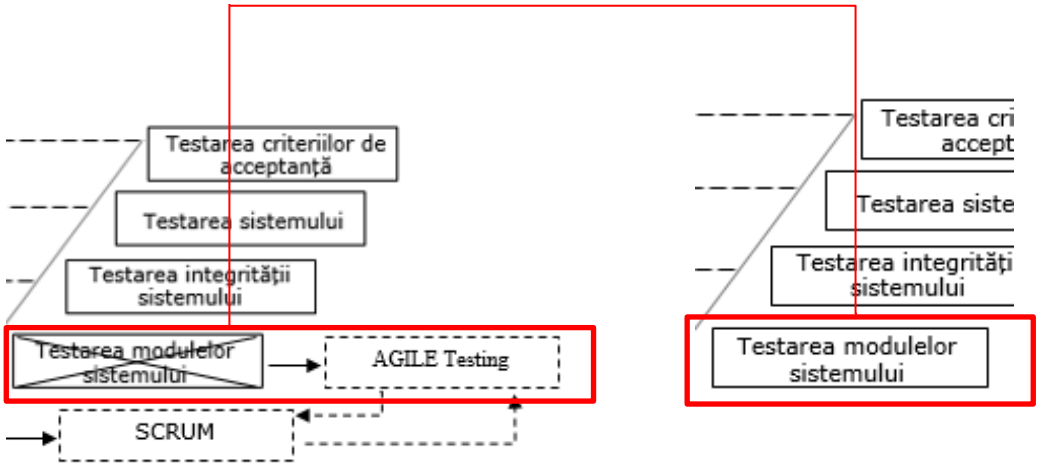

Fig. 5. 7 Comparație între fazele de testare a modulelor sistemului a modelelor 2JCS și V.

Timpul dedicat testării modulelor sistemului s-a întins pe durata a 13 săptămâni de-a lungul proiectului în cazul companiei V și 15 săptămâni în cadrul companiei A. Acești timpi au fost furnizați de către test-managerii celor două proiecte. Autorul a insistat asupra coordonatorilor de test a companiei A și a companiei V cu câte o întrebare esențială legată de utilizarea metodologii AGILE de testare respectiv a metodelor tradiționale de testare a modulelor sistemului.
A: Care este avantajul aplicării metodologiei SCRUM asupra testării modulelor sistemului?
Răspuns A: identificarea erorilor într-o fază timpurie a proiectului și rezolvarea acestora în cadrul aceluiași sprint.
V: Care este riscul major în cazul testării modulelor sistemului abia după implementarea completă a modului software?
Răspuns V: erorile identificate într-o anumită fază a proiectului nu vor putea fi rezolvate din cauza acumulării de erori din toate fazele testării sistemului.

Experiența în domeniu și în utilizarea ambelor modele de dezvoltare au condus autorul la concluzia că metodologia AGILE SCRUM utilizată în testarea modulelor sistemului nu economisește timp direct în aceasta fază ci indirect pe toată durata proiectului. Prin utilizarea metodologiei AGILE, timpul investit în implementarea unei funcționalități nu va duce la îndepărtarea acesteia din cauza erorilor raportate în acest modul. Mai precis în cazul proiectului companiei V modificările cerințelor și apariția erorilor va duce la necesitatea prioritizării erorilor,

fapt ce va duce în final la îndepărtarea anumitor funcționalități din cauza neîndeplinirii criteriilor de calitate. Avantajul utilizării metodologiei AGILE în faza de testare a modulelor sistemului se cuantifică prin obținerea rezultatelor scontate în urma investiției, prin satisfacerea așteptărilor clientului și a utilizatorului final. Orice funcție a sistemului care nu a fost implementată poate duce la disconfortul utilizatorului final și automat la pierderea încrederii în produsul realizat.

Concluzie:

Din analiza comparativă a celor două modele (V și 2JCS) pentru fazele de specificare cerințe, specificare cerințe sistem, proiectare a arhitecturii și proiectare a modulelor sistemului proiectului, respectiv de implementare cod a rezultat că prin aplicarea modelului 2JCS, deși durata de dezvoltare a proiectului rămâne aceeași, gestiunea fondului de timp este îmbunătățită, după cum urmează:

- ***Faza de specificații cerințe are durata mai mică***
- ***Faza de specificații sistem are durata mai mică***
- ***Faza de implementare a codului sursă are durata mai mică***
- ***Fondul de timp economisit la cele trei faze listate mai sus este realocat fazelor de schimbare și implementare a noilor cerințe***

În concluzie, prin aplicarea modelului 2JCS nu se scurtează durata de realizare a proiectului însă se evită întârzierile inerente pe care aplicarea modelului V le înregistrează datorită schimbărilor cerințelor, care apar în generația actuală de proiecte.

5.4 Validarea financiară a modelului 2JCS

Validarea financiară a unui model de implementare software implică analiza pe două axe a acestuia:

- Analiza financiară pe durata derulării proiectului;
- Analiza impactului financiar a rezultatului proiectului.

Fondul de timp economisit în urma aplicării modelului 2JCS în proiectele companiei A atrage după sine, pe lângă câștigul timpului în proiecte și avantaje financiare companiilor. În proiectele companiilor mari se vorbește despre costul orei de lucru, fapt ce generează și suma totală care a fost economisită. Sinteza analizei comparative realizată în cap. 5.3. este prezentată în tabelul 5.2. Această sinteză realizează o imagine clară a repartizării fondului de timp pe fazele principale ale proiectului prin aplicarea modelului în V, a modelului 2JCS, respectiv a fondului de timp economisit prin aplicarea modelului 2JCS.

Se observă că timpul economisit prin aplicarea modelului 2JCS este de 11 săptămâni în faza de specificare a cerințelor și de -2 săptămâni în faza de testare a modulelor sistemului. Din cauză că ora de lucru este renumerată diferit în diferitele faze ale proiectului, economia (C) adusă proiectului pentru fiecare faza este dată de următorii factori:

- numărul de persoane care lucrează în această fază (NPF) ;
- numărul de ore economisite în urma aplicării modelului 2JCS (NOE2);

- valoarea orei de lucru a specialiștilor acestei faze(VS), unde

C1=NPF*NOE2*VS; (5.1)

Tabel 5. 2 Timp economisit pe fiecare fază a proiectului.

Faza a proiectului	**Timp necesar parcurgerii fazei a modelului în V**	**Timp necesar parcurgerii fazei a modelului 2 JCS**	**Diferența timp intre modelele utilizate**
Specificarea cerințelor	53 săptămâni	42 săptămâni	11 săptămâni
Specificarea arhitecturii sistemului	18 săptămâni	~18 săptămâni împărțite pe durata proiectului	0 săptămâni
Proiectarea modulelor software	3saptamani	~ 3 săptămâni împărțite pe durata proiectului	0 săptămâni
Implementarea codului sursa	18 luni	18 luni	0 săptămâni
Testarea modulelor sistemului	13 săptămâni	15 săptămâni	-2 săptămâni
Integrarea sistemului	NA	NA	NA
Testarea sistemului	NA	NA	NA
Testarea criteriilor de acceptanta	NA	NA	NA

În tabelul 5.3 sunt sistematizate informații cu caracter financiar furnizate de firma V și firma A. În scopul validării financiare se pleacă de la următorul calcul economic:

- în faza de specificare de cerințe sunt implicați 5 specialiști.
- numărul de ore economisite sunt 440 de ore, ceea ce corespunde celor 11 săptămâni economisite rezultate conform tabelului 5.2.
- rata standard / ora pentru un specialist este de 80€.

Plecând de la formula 5.1 rezultă un câștig în faza de specificare a cerințelor de 176000€, sumă compusă din numărul de specialiști implicați în această fază (5)* timpul economisit (440 ore) * valoarea orei de lucru (80€). Totodată pierderea provocată în faza de testare a modulelor sistemului este de 40000€, sumă compusă din numărul de specialiști implicați în această fază (10)*timpul economisit (-80)*valoarea orei de lucru (50€).

În concluzie, câștigul financiar adus proiectului aplicând această logică și modelul 2JCS este conform formulei 5.2

C2= 5*11*40*80 – 10* 2*40*50=176000 – 40000 = 136000€, unde:

$$C2 = \sum_{k=1}^{număr\ faze} (NPF * NOE2 * VS) \quad (5.2)$$

Totuși calculul de mai sus nu corespunde realității. Durata proiectelor automotive este fixă, fapt determinat de modificările de cerințe. Proiectele automotive se confruntă cu problema că nu reușesc să se încadreze în timp, ceea ce a determinat și subiectul tezei.

Tabel 5. 3 Costuri proiect analizate pe faze de proiectare în cadrul companiei V

Faza de dezvoltare	**1**	**2**	**3**	**4 = 1*2*3**
	Ore necesare rulare faza de dezvoltare	**Nr. Specialiști**	**Rata standard /ora (€/ora)**	**Val. monetara (€)**
Specificare Cerințe	2120	5	80	848000
Specificare Arhitectura Sistem	720	4	100	288000
Proiectarea Modulelor Sistemului	120	10	90	108000
Implementarea Codului Sursa	2880	250	80	57600000
Testarea modulelor	520	10	50	260000

Conform tabelului 5.4 aplicând modelul 2JCS pe fazele inițiale ale proiectului reiese o pierdere, însă în tabelul 5.3 și tabelul 5.4 nu sunt incluse fazele de implementare a schimbărilor cerințelor care sunt inerente proiectelor actuale.
Pornind de la obiectivul declarat al acestei teze, acela al gestiunii timpului datorat implementării noilor cerințe, în urma aplicării modelului 2JCS nu mai apar faze intermediare de actualizare a proiectului cum se întâmplă în cazul aplicării modelului în V. În cazul aplicării modelului în V proiectul este afectat de cel puțin o fază intermediară de actualizare a căror costuri sunt prezentate în continuare.

În cazul aplicării modelului 2JCS economia de timp rezultată din faza de specificare a cerințelor va fi utilizată în implementarea codului sursă și acoperirea timpului suplimentar necesar testării modulelor sistemului.

Astfel câștigul financiar rezultat în urma reducerii cu 9 săptămâni(tabelul 5.2) a timpului proiectului va fi redus de necesitatea implementării cerințelor modificate (Fig. 5.8). Deoarece ora de lucru a specialiștilor diferă de la o fază la alta, câștigul financiar se calculează conform tabelului 5.3 și tabelului 5.4.

Tabel 5. 4 Costuri proiect analizate pe faze de proiectare în cadrul companiei A

Faza de dezvoltare	1	2	3	4 = 1*2*3	Câștig/ pierdere
	Ore necesare rulare faza de dezvoltare	Nr. Specialiști	Rata standard /ora (€/ora)	Val. monetara (€)	
Specificare Cerințe	1680	5	80	672000	+176.000
Specificare Arhitectura Sistem	720	4	100	288000	0
Proiectarea Modulelor Sistemului	120	10	90	108000	0
Implementarea Codului Sursa	3240	250	80	64800000	-7.200.000
Testarea modulelor	600	10	50	300000	-40000
Total					-7.064.000

În etapa de implementare a schimbărilor cerințelor autorul a analizat costurile și impactul rezultatului proiectului ca urmare a utilizării celor două modele de dezvoltare. În cadrul acestei etape aplicarea modelului tradițional în cadrul companiei V a dus la imposibilitatea de implementare finală a tuturor cerințelor specificate inițial. Acest lucru s-a datorat necesității implementării cerințelor modificate precum și inflexibilitatea modelului la schimbări. Concret, în cazul companiei V, orice modificare de cerință a dus la excluderea altor funcționalități sau nerezolvarea anumitor erori ale sistemului, costurile rezultate fiind portate într-un nou proiect de actualizare a sistemului actual. Astfel, pe lângă costurile concrete generate de implementarea cerințelor, s-au generat noi costuri prin deschiderea unui nou proiect. Numărul modificărilor cerințelor aduse proiectului în cadrul companiei V a fost de 73, în valoare medie de aproximativ 100.000€. Valoarea cerințelor portate a fost apropiată ca valoare cu valoarea cerințelor noi care au fost implementate, pe lângă costul efectiv de implementare al cerințelor portate, au fost generate costuri mari legate de deschiderea unui nou proiect, valoarea unei noi versiuni a proiectului fiind de aproximativ 10.000.000 €. Costurile totale se compun din:

$$C_{total} = \sum_{k=1}^{număr\ departamente} \text{Costuri\_departamente} \quad \text{, unde:} \qquad (5.3)$$

Costuri_departamente reprezintă costurile fiecărui departament care va fi implicat în desfășurarea noului proiect. Pe lângă departamentul de dezvoltare care va fi nevoit să participe la proiect, următoarele departamente vor genera costuri din cauza efortului care va fi depus de acestea:

- departamentul de marketing;
- departamentul de testare;

- departamentul de calitate;
- departamentul de testare intensivă de lungă durată;
- departamentul responsabil de coordonarea proiectului general din care face parte proiectul de radio-navigație.

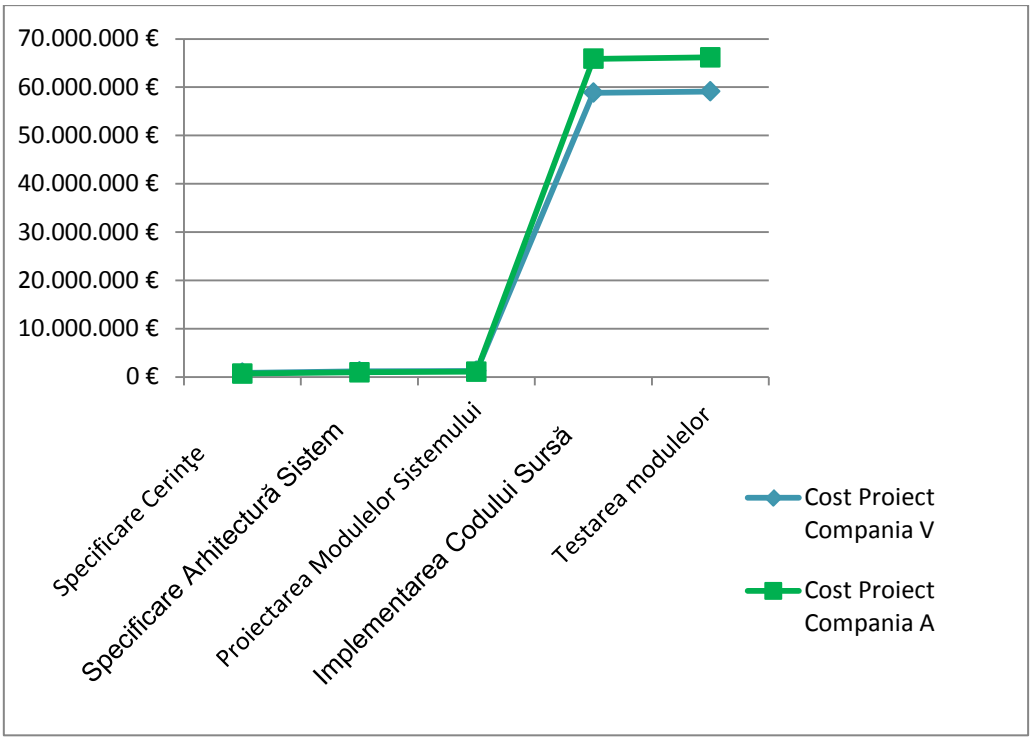

Fig. 5. 8 Evoluția comparativa a costului proiectului

Fig. 5.9 prezintă numai costurile generate de echipa de dezvoltare a proiectului în fazele diferite față de modelul 2JCS din cauza portării anumitor cerințe pe un proiect de actualizare ulterior.

Pornind de la faptul că analiza comparativă a proiectului derulat în compania A cu cel derulat în compania V au fot aproape identice, cerințele de actualizare a celor două proiecte derulate în paralel au fost aceleași. În cazul companiei A, modelul de dezvoltare 2JCS, prin intermediul modelelor JAD a facilitat implementarea cerințelor de actualizare pe parcursul dezvoltării curente a cerințelor inițiale a proiectului. Astfel rezultând doar 40 de noi modificări. Modelul în V, fiind foarte rigid nu a permis nici o actualizare pe parcursul derulării proiectului, astfel încât au rezultat în cadrul proiectului de portare 73 de modificări. Numărul redus de cereri de modificare(40) a cerințelor și avantajele utilizării metodologiei SCRUM în faza de implementare a codului sursă au condus la facilitarea implementării acestor modificări fără să mai fie necesară portarea cerințelor prin intermediul unui nou proiect de actualizare așa cum s-a realizat în cadrul companiei V.

În fig. 5.9 sunt configurate costurile de portare conform formulei 5.3.

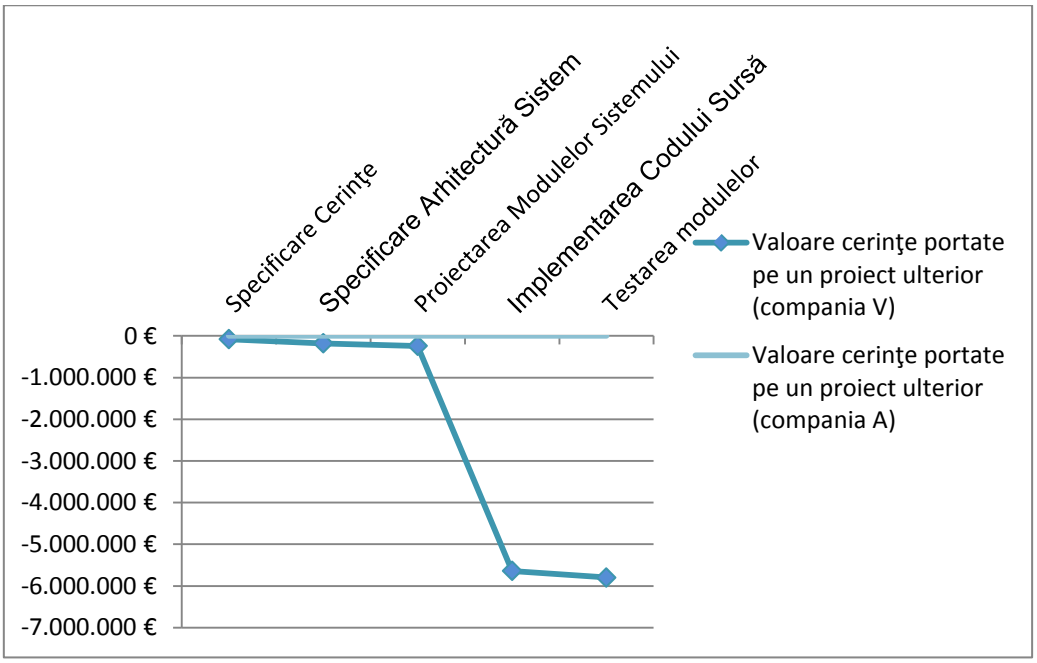

Fig. 5. 9 Costuri generate proiectelor pe diferitele faze ale proiectelor companiilor A și V

Concluzionând, implementarea unui proiect automotive utilizând modelul 2JCS aduce economii companiei pe durată medie. Acestea se compun din:

- costurile economisite ale proiectului, datorate neportării cerințelor într-un alt proiect;
- câștigul reputației companiei ca fiind de încredere.

5.5 Concluzii

Dezvoltarea de noi modele de dezvoltare software aduce după sine necesitatea validării comparative a acestora. Criteriile principale de validare sunt inspirate din criteriile principale de succes ale proiectelor, performanța timpului și cea financiară.

În urma validării modelului 2JCS pot fi subliniate următoarele concluzii:

- din cauza structurii proiectelor automotive în care se va folosi modelul 2JCS, acesta nu va aduce economii de timp, ci posibilitatea de implementare a unei cantități mai mare de cerințe actualizate, fără ca proiectul inițial să fie întârziat;
- din punct de vedere financiar utilizarea modelului 2JCS va aduce economii companiilor pe termen mediu, datorită facilitării respectării duratei de finalizare a proiectelor în cazul apariției cerințelor de actualizare.

Firma A în care a fost implementat modelul 2JCS a obținut următoarele avantaje în comparație cu firma V:

- Stabilirea clară și concisă a naturii și concesiunii activităților și fazelor care alcătuiesc faza de actualizare .
- Odată cu respectarea duratei de finalizare a proiectului în firma A, s-a obținut o mai bună organizare a echipelor de lucru, acestea fiind continuu pregătite pentru formularea de noi cerințe și reintegrarea cerințelor inițiale care să permită inserarea de noi funcționalități.
- Prin respectarea duratei de finalizare a proiectului, respectiv prin integrarea noilor funcționalități în paralel cu cele întâile, proiectul din firma A a fost dezvoltat cu deviații minore de la bugetul inițial aprobat.
- Chiar dacă anumite faze de dezvoltare au depășit bugetul declarat inițial, s-a asigurat un beneficiu, datorită încadrării proiectului în durata inițială de timp și datorită scăderii cuantumului costurilor prin neinițierea unui proiect nou de portare a cerințelor de actualizare (cum se realizează în mod inerent prin aplicarea modelul în V).
- Utilizarea modelului 2JCS va aduce economii companiilor pe termen mediu, datorită facilitării respectării duratei de finalizare a proiectelor în cazul apariției cerințelor de actualizare.

Contribuțiile personale ale autorului sunt:

- analizarea posibilităților de validare a modelului 2JCS prin studierea literaturii de specialitate;
- sinteza criteriilor pe care trebuie să le îndeplinească un model de dezvoltare software;
- culegerea informațiilor și datelor necesare derulării proiectelor software de radio-navigație din firma A, respectiv firma V;
- analiza comparativă a dezvoltării celor două proiecte în firmele A și V pe baza cărora s-a putut realiza validarea modelului 2JCS;
- conceperea procesului de validare din perspectiva încadrării proiectelor de radio-navigație în eșalonarea calendaristică inițială, indiferent de variațiile specificațiilor inițiale.
- conceperea procesului de validare din perspectiva eficienței economice, prin prisma combinației celor două variabile economice „cost” și „timp” determinată de aplicarea modelului 2JCS.
- analiza comparativă a modelelor tradiționale de dezvoltare și a modelului 2JCS din punct de vedere financiar;
- analiza comparativă a modelelor tradiționale de dezvoltare și a modelului 2JCS din punct de vedere temporal;
- prezentarea modului de distribuție a costurilor de dezvoltare a proiectelor.

6. CONCLUZII, CONTRIBUŢII PERSONALE ȘI DIRECŢII VIITOARE DE DEZVOLTARE

6.1 Concluzii

Actualitatea problematicii abordate în teza de față rezidă din faptul că metodele și tehnicile generate de modelul de dezvoltare în V pentru managementul proiectului în cadrul industriei automotive au devenit prea rigide pentru contextul actual, exprimând-se în mod continuu capacitatea scăzută a tuturor tipurilor de resurse utilizate în astfel de proiecte. Efectul asupra renumelui companiilor producătoare de noi tehnologii au determinat necesitatea cercetării de noi metode de dezvoltare a managementului de proiect, care să poată fi aplicate în proiectele de tip automotive. Presiunea crescândă în proiectele din domeniul automotive pentru respectarea „pietrelor de hotar" au dus într-o primă fază la analiza proceselor folosite în dezvoltarea sistemelor de radio-navigație. În urma identificării slăbiciunilor modelelor consacrate, utilizate în domeniul automotive, autorul a conceput un nou model de dezvoltare (2JCS), care să faciliteze eliminarea neconcordantelor care se exprimă în mod curent în managementul actual al proiectelor de tip automotive. Utilizarea noului model permite economisirea duratelor de timp din proiecte, aducând pe termen lung și avantaje financiare.

Conceperea unui model de dezvoltare(2JCS) pentru managementul proiectelor de tip automotive nu ar avea nici o valoare dacă acesta nu ar putea fi validat pe cele două axe, a timpului și cea financiară. În acest scop a fost prezentată în lucrare comparația între modelele consacrate de dezvoltare din domeniul automotive și noul model de dezvoltare conceput.

Complexitatea sistemelor de radio-navigație în combinație cu procesele și metodele folosite în implementarea acestora au determinat obiectivele lucrării de față.

În contextul obiectivului principal declarat al tezei, se pot evidenția următoarele obiective intermediare abordate în teză:

- Definirea complexă a managementului de proiect pornind de la istoricul acestuia până la definirea managementului de proiect pentru sistemele de radio-navigație;
- Analiza avantajelor și dezavantajelor utilizării ciclurilor de viață tradiționale și a metodologiilor AGILE în proiecte;
- Analiza impactului utilizării teoriei constrângerii dezvoltata de E.M. Goldratt aplicată în proiectele automotive;
- Analiza impactului utilizării teoriei hazardului (HAZOP) în proiectele automotive;
- Validarea modelului conceput (2JCS) prin analiza comparativă a modelului tradițional cu noul model implementat în dezvoltarea proiectelor automotive.

Având la bază obiectivele declarate în cadrul acestei teze, în continuare se pot evidenția ca principale concluzii următoarele:

- atingerea obiectivelor proiectelor presupune cunoașterea detaliată a fazelor, pașilor proiectelor, și a ordinii în care aceștia trebuie să se desfășoare, precum și a factorilor critici specifici tipului de proiect;
- înțelegerea importanței factorilor critici stă la baza succesului proiectului, în domeniul automotive acest lucru fiind posibil având în vedere faptul că literatura de specialitate abordează profund aceste aspecte;
- standardizarea proceselor utilizate în proiectele automotive, sub forma proceselor Spice, ajută coordonatorii de proiect în organizarea și luarea deciziilor pentru metodele de control și comandă a acestor tipuri de proiecte;
- ciclurile de viață tradiționale, utilizate în managementul proiectelor software nu diferă în esență, acestea deosebindu-se doar în detaliu;
- cel mai des utilizat model în proiectele de tip automotive este modelul în „V", acesta fiind o abordare superioară a altui model tradițional, denumit „în cascadă";
- utilizarea metodelor de dezvoltare tradiționale în proiectele automotive este determinată de motive istorice, mai concret, se pleacă de la faptul că producția în acest domeniu este dezvoltată conform modelului în „V", proiectele preluând de la sine aceeași configurație de cicluri;
- utilizarea metodelor de dezvoltare tradiționale în producție, metode care stau și la baza dezvoltării proiectelor software automotive au la bază modele, care nu permit modificarea semnificativă a cerințelor de-a lungul dezvoltării proiectelor;
- inflexibilitatea modelelor tradiționale de dezvoltare în cadrul proiectelor de dezvoltare software au condus la integrarea metodologiilor AGILE;
- metodologiilor AGILE reușesc să elimine slăbiciuni ale metodelor tradiționale prin accentul pus pe comunicarea și armonizarea echipelor ori de câte ori apare o perturbație în proiect, introducând-se astfel o reală flexibilitate în abordarea schimbărilor;
- modificarea sau schimbarea cerințelor, în orice fază a ciclului de dezvoltare a proiectului reprezintă deja o certitudine în proiectele de dezvoltare software din industria automotive;
- metodele moderne ca de ex. „Teoria Constrângerilor" dezvoltată de E.M. Goldratt oferă explicații si soluţii asupra neajunsurilor proceselor tradiționale în dezvoltarea software, oferind soluţii care să facă faţă strategiilor de viteză însă, nu poate rezolva în totalitate integrarea schimbărilor survenite pe parcursul ciclului de viaţă al proiectului;
- explicația asupra neajunsurilor proceselor tradiționale în dezvoltarea software poate fi nuanțată prin sinteza factorilor de risc și decizionali care duc la implementarea unei cerințe noi sau modificate cu scopul propunerii unei soluții astfel încât calendarul proiectului să nu fie influențat;
- din cauza structurii proiectelor automotive în care se va folosi modelul 2JCS, acesta nu va aduce economii de timp, ci posibilitatea de implementare a unei cantități mai mare de cerințe actualizate, fără ca proiectul inițial să fie întârziat;
- din punct de vedere financiar utilizarea modelului 2JCS va aduce economii companiilor pe termen mediu, datorită facilitării respectării

- duratei de finalizare a proiectelor în cazul apariției cerințelor de actualizare.

6.2 Contribuțiile personale ale autorului

Contribuțiile personale ale autorului au fost descrise în fiecare capitol din cadrul lucrării, fiind sintetizate în tabelele 6.1, 6.2 și 6.3:

Tabel 6. 1 Contribuții teoretice

Nr. crt.	Descrierea contribuției	Cap.
1	Sinteza fazelor reprezentative proiectelor în general, cu particularizări în proiectele automotive;	2
2	Sinteza factorilor critici reprezentativi proiectelor, pe baza unei analize comparative din literatura de specialitate;	2
3	Descrierea factorilor critici reprezentativi proiectele automotive, precum și prezentarea factorilor critici specifici proiectelor automotive.	2
4	Analizarea caracteristicilor principale ale metodelor tradiționale de dezvoltare a proiectelor.	3
5	Analiza principalelor metodologii AGILE care vin să elimine slăbiciuni ale metodelor tradiționale.	3
6	Analiza și descrierea concluziilor literaturii de specialitate legate de stabilitatea cerințelor în proiectele software.	4
7	Sinteza criteriilor pe care trebuie să le îndeplinească un model de dezvoltare software.	5
8	Analizarea posibilităților de validare a modelului 2JCS prin studierea literaturii de specialitate.	5

Tabel 6. 2 Contribuții teoretice aplicate în practică

Nr. crt.	Descrierea contribuției	Cap.
1	Identificarea unui nou factor critic reprezentativ proiectelor automotive, mai concret acela al produsului inovativ, care începe să aibă o pondere prioritară față de cei clasici, considerați până acum ca reper absolut: timp, cost și calitate.	2
2	Justificarea utilizării modelelor de dezvoltare în V în proiectele software din domeniul automotive.	3
3	Analiza și sinteza comparativă între procesele de dezvoltare tradiționale vs. metodologiile AGILE.	3
4	Analiza și sinteza comparativă între efectele modificării cerințelor în funcție de stadiul de dezvoltare a proiectului.	3
5	Prezentarea impactului modificărilor asupra calendarului proiectului.	3
6	Identificarea necesității dezvoltării unui nou model de dezvoltare specific proiectelor de tip automotive de mari dimensiuni.	3

7	Conceperea unei metode de decizie bazată pe formula matematica asupra oportunității implementării cerințelor noi sau modificate în dezvoltarea proiectelor de tip automotive;	4
8	Analiza proceselor de dezvoltare automotive și influența proceselor de producție asupra acestora;	4
9	Analiza și descrierea teoriei constrângerilor cu scopul validării noului model de dezvoltare;	4
10	Analiza și elaborarea factorilor care determină implementarea cerințelor noi;	4
11	Culegerea informațiilor și datelor necesare derulării proiectelor software de radio-navigație din firma A, respectiv firma V;	5
12	Analiza comparativă a dezvoltării celor două proiecte în firmele A și V pe baza cărora s-a putut realiza validarea modelului 2JCS;	5

Tabel 6. 3 Contribuții practice

Nr. crt.	**Descrierea contribuției**	**Cap.**
1	Configurarea unui nou model de dezvoltare, flexibil la schimbări, cu scopul de atenuare a efectului schimbării cerințelor în dezvoltarea proiectelor de tip automotive (**modelul 2JCS**);	4
2	Analiza comparativă a modelelor tradiționale de dezvoltare și a **modelului 2JCS** din punct de vedere și temporal;	5
3	Contribuţia esenţială a **modelului 2JCS** este aceea că permite implementarea unei cantități mai mari de cerințe actualizate, fără ca proiectul inițial să fie întârziat;	5
4	Analiza comparativă a modelelor tradiționale de dezvoltare și a modelului 2JCS din punct de vedere financiar şi temporal;	5
5	Utilizarea modelului 2JCS va aduce economii companiilor pe termen mediu, datorită facilitării respectării duratei de finalizare a proiectelor în cazul apariției cerințelor de actualizare;	5
6	Prezentarea modului de distribuție a costurilor de dezvoltare a proiectelor;	5
7	Conceperea procesului de validare din perspectiva încadrării proiectelor de radio-navigație în eșalonarea calendaristică inițială, indiferent de variațiile specificațiilor inițiale.	5
8	Conceperea procesului de validare din perspectiva eficienţei economice, prin prisma combinației celor două variabile economice „cost” și „timp” determinată de aplicarea modelului 2JCS.	5

6.3 Direcții de dezvoltare viitoare

Complexitatea și inflexibilitatea proiectelor de radio-navigație din domeniul automotive a impus crearea unui nou model de dezvoltare. Conceperea noului model (2JCS) deschide noi orizonturi cercetării:

- Extinderea lucrării și a modelului 2JCS prin analiza și înlocuirea fazelor de testare tradiționale din cadrul modelului cu fazele de testare a altor modele sau metodologii;

- Dezvoltarea unei modalități de validare a modelelor de dezvoltare utilizate în proiectele software;
- Dezvoltarea rapidă a tehnologiilor vor genera noi probleme care trebuie soluționate aplicând noi procese în proiectele automotive. Modificări ale mediului proiectului sau ale factorilor critici ale proiectului vor impune adaptarea modelului 2JCS astfel încât utilizarea acestuia să ducă la succesul proiectului.

Anexe

A1. LISTĂ DE LUCRĂRI PUBLICATE ÎN DOMENIUL TEZEI DE DOCTORAT

Nr. crt.	Autori	Denumirea lucrării	Denumirea conferinței unde a fost susținută (link la site-ul conferinței)	Locul și intervalul în care a avut loc conferința
1	Andrei Hutanu, Gabriela Prostean, Dumitru Mnerie, Andra Badea	Research of Requirements Constraints in Automotive Projects	CENTERIS 2013 - Conference on ENTERprise Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies	Noiembrie 2013, Lisabona, Portugalia
2	Andrei Hutanu, Gabriela Proștean, Alin Vasile Mnerie, Raluca Schiopu	Contemporaneous Issues in e-Learning Projects along European Union	6th World Conference on Educational Sciences, WCES 2014	6-9 Februarie 2014, Malta
3	Andrei Hutanu, Gabriela Prostean, Dumitru Mnerie,Andra Badea	Integrating Critical Chain Method with AGILE Life Cycles in the Automotive Industry	7th World Conference on Educational Sciences, WCES 2015	5-7 Februarie, 2015 Atena, Grecia

4	Andrei Hutanu, Gabriela Prostean, Stephan Volker, Dumitru Mnerie	Opportunity Analysis of Change Requests in Automotive Projects	6th International Workshop on Soft Computing Applications, SOFA 2014 http://sofa2014.org/	24-26 Julie 2014, Timişoara, România
5	Andra Badea, Gabriela Proștean, Andrei Hutanu, Serban Popa	Competency training în collaborative supply chain using KSA model	6th World Conference on Educational Sciences, WCES 2014	6-9 Februarie 2014, Malta
6	Stephan Volker, Gabriela Prostean, Andrei Hutanu	Research of Automotive Change Management and supportive Risk-Management	6th International Workshop on Soft Computing Applications, SOFA 2014 http://sofa2014.org/	24-26 Julie 2014, Timişoara, România
7	Gabriela Prostean, Andrei Hutanu, Stephan Volker	Impact of agile methodologies on team capacity in automotive radio-navigation projects	International Conference on Applied Sciences ICAS2016	25-27 Mai, 2016, Hunedoara, România
8	Gabriela Prostean, Stephan Volker, Andrei Hutanu	Change Management Methodologies trained for Automotive Infotainment Projects	International Conference on Applied Sciences ICAS2016	25-27 Mai, 2016, Hunedoara, România

8	Dumitru Mnerie, Andrei Hutanu, Catalin Crisan	CONSIDERATIONS ON SOME KEY FACTORS FOR THE PROGRESS OF THE UNCONVENTIONAL TECHNOLOGIES	Revista de Tehnologii Neconventionale16.2 (Iunie 2012)	Iunie, 2012, Sibiu, România
9	Alin Vasile Mnerie, Dumitru Mnerie, Andrei Hutanu, Mihai Condescu	Global analysis of cutting system by electrical erosion with wire	Revista de Tehnologii Neconventionale17.2 (Iunie 2013)	Iunie, 2012, Sibiu, România
10	Mnerie, Alin Vasile; Mnerie, Dumitru; Hutanu, Andrei;	WIRED ELECTRICAL EROSION OF HARD AND SUPERHARD METAL COMPONENTS OF AGRICULTURAL EQUIPMENT	Proceedings of the 42nd International Symposium on Agricultural Engineering, Actual Tasks on Agricultural Engineering,	25-28 Februarie, 2014, Opatija, Croatia

7. BIBLIOGRAFIE

[1] A. Badea, G. Prostean, A. Hutanu, 2014, Competency Training in Collaborative Supply Chain Using KSA Model, Procedia - Social and Behavioral Sciences, Volum 191, paginile 500-505; ISSN: 1877-0428

[2] A. Hutanu, G. Prostean, D. Mnerie, R. Schipu, 2014, Contemporaneous Issues în e-Learning Projects Along European Union.

[3] A. Hutanu, G. Prostean, A. Badea, 2015, Integrating Critical Chain method with AGILE life cycles in the automotive industry

[4] Agarwal, N. & Rathod, U. 2006, Defining success for software projects: An exploratory revelation, International Journal of Project Management, Vol. 24, pp. 358–370.

[5] Andrei Hutanu, Gabriela Prostean, Dumitru Mnerie, Andra Badea, 2013, Research of Requirements Constraints în Automotive Projects.

[6] Andrei Hutanu, Gabriela Prostean, Dumitru Mnerie, Andra Badea (2014). Opportunity Analysis of Change Requests în Automotive Projects.

[7] Armstrong, S. (2001), 2001 Principles of forecasting: a handbook for researchers and practitioners

[8] Atkinson R. Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. Int J Project Manage 1999;17(6):337–43.

[9] B.C.D. Anda, D.I.K. Sjøberg, A. Mockus 2009, Variability and reproducibility în software engineering: a study of four companies that developed the same system

[10] Regnell, S. Brinkkemper, 2005, Market-driven requirements engineering for software products, în: A. Aurum, C. Wohlin (Eds.), Engineering and Managing Software Requirements, Springer, Berlin Heidelberg, pp. 287–308.

[11] Belassi, W. and O.I. Tukel,1996, A new framework for determining critical success/failure factors în projects. International Journal of Project Management

[12] Belout, A. & Gauvreau, C. (2004), Factors influencing project success: the impact of human resource management

[13] Benediktsson O, Dalcher D, Thorbergsson H (2006) Comparison of software development life cycles: a multiproject experiment.

[14] Brown SL, Eisenhardt KM, 1997. The art of continuous change: linking complexity theory and time-paced evolution în relentlessly shifting organization.

[15] Hood, S. Wiedemann, S. Fichtinger, U. Pautz, 2008, Requirements Management The Interface Between Requirements Development and All Other Systems Engineering Processes, Springer, Berlin

[16] Chung-Suk Cho and G. Edward Gibson Jr., 2001, Building Project Scope Definition Using Project Definition Rating Index Members,ASCE

[17] Clarke, A. 1999, A Practical use of key success factors to improve the effectiveness of project management, International Journal of Project Management, Vol. 17, pp. 139-145

[18] Cockburn, A., 2002, Agile Software Development. Boston, Addison-Wesley

[19] Cockburn A., 1998, Surviving Object-Oriented Projects: A Manager's Guide. Addison Wesley Longman.

[20] Cohn, M., Ford, D., 2003. Introducing an agile process to an organization. Computer 36 (6), 74–7

[21] Collins, A. & Baccarini, D. 2004, Project Success - A Survey, Journal of Construction Research, Vol. 5, No. 2, pp. 211-231

[22] Cooke-Davies, T. 2002, The real success factors on projects, International Journal of Project Management, Vol. 20, No. 3, pp. 185-190.

[23] Cooper RG. 2001, Winning at new products. Reading, MA: Perseus Books.

[24] D Pfahl, K Lebsanft, 2000, Using simulation to analyse the impact of software requirement volatility on project performance.

[25] D Zowghi, N Nurmuliani, 2002 A study of the impact of requirements volatility on software project performance

[26] Dulce Domingos, Ricardo Martinho, Carlos Cândido, Vladimir Modrák, 2013, Flexibility în cross-organizational WS-BPEL business processes

[27] Dvir, D. Lipovetsky, S. Shenhar, A. & Tishler, A., 1998, In search of project classification: a non-universal approach to project success factors

[28] Dvir, D. Raz, T. & Shenhar, A, 2003, An empirical analysis of armthe relationship between project planning and project success

[29] Eisenhardt K. 1989. Making fast strategic decisions în high-velocity environments. Acad Manage J;32(2):543–576

[30] Eisner, E. W. (1991). The enlightened eye: Qualitative inquiry and the enhancement of educational practice. Upper Saddle River, NJ, Prentice Hall.

[31] Els Du Bois, Imre Horvath, Operationalization of the quadrant-based validation in case of a designerly software development methodology, Design Methods and Tools

[32] Eppler, M., 2003, Managing information quality: Increasing the value of information în knowledge-intensive products and processes. Berlin,Germany: Springer-Verlag.

[33] Ernest Mnkandla, 2008, A SELECTION FRAMEWORK FOR AGILE METHODOLOGY PRACTICES: A Family of Methodologies Approach

[34] G. Avram, 2013, Metodologii de dezvoltare a programelor informatice – studiu comparativ.

[35] Gabriela Fernandes, Stephen Ward, Madalena Araújo 2013, Developing a Framework to Improve and Embed Project Management Practices în Organisations

[36] GE Stark, P Oman, A Skillicorn, A Ameele , 1999, An examination of the effects of requirements changes on software maintenance releases

[37] Goldratt, E. M., 2004, The Goal, the North River Press Publishing Corporation, Great Barrington.
[38] Goldratt, E.M., 1997, Critical Chain, the North River Press Publishing Corporation, Great Barrington.
[39] Gottschalk, P. (2002). "Empirical Validation Procedure for the Knowledge Management Technology Stage Model." Informing Science The International Journal of an Emerging Transdiscipline 5(4): 30.
[40] H Takeuchi, I Nonaka 1986 "New Product Development Game". Harvard Business Review 86116:137–146.
[41] Hall, T., Beecham S, Verner J., and Wilson D., 2008, “The impact of staff turnover on software projects: the importance of understanding what makes software practitioners tick” ACM SIGMIS Conference, 2008.
[42] http://agilemanifesto.org/
[43] http://agilemodeling.com/essays/amdd.htm. Accesat în 21.09.2014
[44] http://www.automotivespice.com/ Accesat in 24.10.2015
[45] http://pm4id.org/1/1/ Accesat in 24.10.2015
[46] http://project-management.com/top-5-project-management-phases/ Accesat in 24.10.2015
[47] http://de.slideshare.net/HassnainJamil/4-gspm, Accesat in 05.10.2015
[48] http://seor.gmu.edu/insert/robot/robot2.html, Accesat in 05.10.2015
[49] http://www.coleyconsulting.co.uk/testtype.htm, Accesat in 05.10.2015.
[50] http://www.extremeprogramming.org/map/project.html. Accesat în 21.09.2014
[51] http://www.google.de/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&ved=0CDkQFjADahUKEwiPns6b9NvIAhVIVhQKHRftCNA&url=http%3A%2F%2Fgunston.gmu.edu%2Fhealthscience%2FProjectManagementInIT%2FSlides%2FHistoryOfProjectManagementPartOne.ppt&usg=AFQjCNHn1cjbJhbKlb6ws4f0M5FuSGu7Rg&bvm=bv.105841590,d.bGQ&cad=rja Accesat in 24.10.2015
[52] http://www.gpm-ipma.de/qualifizierung_zertifizierung/ipma_4_l_q_lehrgaenge_fuer_projektmanager/ ipma_competence_baseline_icb.html Accesat in 24.10.2015
[53] http://www.instructionaldesign.org/models/addie.html. Accesat in data de 19.09.16.
[54] http://knowledge.apm.org.uk/bok/life-cycle, Acesat in 05.10.2015
[55] http://www.lifecyclestep.com/browse/411.1.4JADSessions.htm, accesat la 25.12.15
[56] http://www.methodpark.de/praesentationen/praesentationen/download-details.html?tx_abdownloads_pi1[uid]=47
[57] http://www.projectsmart.co.uk/brief-history-of-project-management.php, accesat pe 25.03.15
[58] http://www.techrepublic.com/article/jad-sessions-will-speed-up-the-project-definition-process/. Accesat la 25.12.15.
[59] http://www.umsl.edu/~sauterv/analysis/488_f02_papers/ProjMgmt.html Accesat in 24.10.2015

[60] http://www.w3computing.com/systemsanalysis/joint-application-design, accesat la 25.12.15.
[61] https://www.apm.org.uk/event/what-romans-project-managed-us Accesat in 24.10.2015
[62] https://www.me.utexas.edu/~ppmdlab/files/ccs.valid.square.Jan05.pdf Accesat in 26.02.2016.
[63] https://www.prince2.com/eur/what-is-prince2 Accesat in 24.10.2015
[64] https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf#zoom=100). Accesat în 21.09.2014
[65] Hughes, S. W. Tippett, D. D. & Thomas, W. K., 2004, Measuring Project Success in the Construction Industry.
[66] J. A. Highsmith (2000). Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, New York: Dorset
[67] John S. Carson, II (2002), Model verification and validation. 37th Winter Conference on Simulation. Orlando, Florida: 130-143.
[68] Jordi Dunjó, Vasilis Fthenakis, Juan A. Vílchez, Josep Arnaldos, 2010, Hazard and operability (HAZOP) analysis. A literature review, Journal of Hazardous Materials, Volume 173, Issues 1–3, 15 January 2010, Pages 19–32
[69] K. de Bakker, A. Boonstra, H. Wortmann, 2010, Does risk management contribute to IT project success? A meta-analysis of empirical evidence
[70] K. El Emam, A.G. Koru, 2008, A replicated survey of IT software project failures
[71] K Schwaber, M Beedle, 2002, Software Development with Scrum
[72] K.T. Yeo, 2002, Critical failure factors in information systems projects
[73] Kerzner, H. 2006, Project Management Best Practices: Achieving Global Excellence, New York.
[74] Kerzner H. Applied project management. Best practices on implementation. New York: John Wiley, 2000.
[75] Kitchenham et al., 1997, DESMET: a methodology for evaluating software engineering methods and tools. Computing & Control Engineering Journal 8(3): 120-126
[76] Klaus Hoermann, Markus Mueller, Lars Dittmann, Joerg Zimmer, 2008, Automotive Spice în Practice
[77] L.A. Kappelman, R. McKeeman, L. Zhang, 2006,Early warning signs of IT project failure: the dominant dozen
[78] L.S.L. Lai, 1997, A synergistic approach to project management in information systems development
[79] Larman, C., 2004, Agile and Iterative Development: A Manger's Guide, Addison– Wesley, USA, pp. 9–39.
[80] Lawley, H.G., Operability studies and hazard analysis, Chemical Engineering Progress 70 (4), 1974
[81] Leidecker JK, Bruno AV 1984, Identifying and using critical success factors. Long Range Planning

[82] Lim, C. S. & Mohamed, M. Z. (1999), Criteria of project success: an exploratory re-examination

[83] Linberg, K. R. (1999), Software developer perceptions about software project failure: a case study

[84] Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., et al., 2004. Agile software development in large organizations. Computer 37 (12), 26–3

[85] M. DeBellis, C. Haapala, 1995, User-centric software engineering, IEEE Exp. 10 (1) 34–41

[86] M. Sumner, D. Bock, G. Giamartino, 2006, Exploring the linkage between the characteristics of it project leaders and project success

[87] M. Weber, J. Weisbrod, 2003, “Requirements Engineering în Automotive Development – Experiences and Challenges”, IEEE Software 20(1).

[88] Macal, C. M. (2005). Model verification and validation. w. i. t. a. s. s. m. a. models". Chicago, IL, center for complex adaptive agent system simulation.

[89] Maurer, F. and Martel, S., 2002, Extreme Programming: Rapid Development for Web–Based Applications, IEEE Internet Computing, pp.86–90.

[90] Martin, R.C. 1999, Iterative and Incremental Development (IID), Engineering Notebook Column, C++ Report

[91] Maxwell, J. A. (1992). "Understanding and validity in qualitative research." Harvard Educational Review 62: 279-300.

[92] Merisalo-Rantanen, H., T. Tuunanen and M. Rossi, 2005, "Is Extreme Programming Just Old Wine în New Bottles: A Comparison of Two Cases".

[93] Milosevic D., Patanakul P., 2005. Standardized project management may increase development projects success, International Journal of Project Management 23(3), p.181-192.

[94] Muhammad Nabeel Mirza, Zohreh Pourzolfaghar, Mojde Shahnazari.2013. Significance of Scope în Project Success.

[95] Munns, A.K. & Bjeirmi, B.F. (1996), The role of project management în achieving project success

[96] Olewnik, A. T. and K. E. Lewis (2003). On validating design decision methodologies. ASME 2003 Design Engineering Technical Conferences - Design Theory and Methodology Conference, Chicago, IL., ASME.

[97] Olsen, R. P. 1971. Can project management be defined?, Project Management Quarterly.

[98] Qumer.,&Sellers, 2007.An evaluation of the degree of agility in six agile methods and its implacability for method engineering. Information and Software Technology 50 (2008) 280-295

[99] Palmer, S. R. and Felsing, J. M. ,2002. A Practical Guide to Feature-Driven Development. Upper Saddle River, NJ, Prentice-Hall

[100] Pedersen at al. (2000), VALIDATING DESIGN METHODS & RESEARCH: THE VALIDATION SQUARE, 2000 ASME Design Engineering Technical Conferences

[101] Pekka Abrahamsson, Outi Salo, Jussi Ronkainen, Juhani Warsta (2002). Agile software development methods. http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf. Accesat în 21.09.2014.

[102] Project Management Body of Knowledge, TenStep Student Handbook, 2005.

[103] Reiss B. 1993, Project Management Demystfied. E and FN Spon, London

[104] Rose, K. 2005, Project Quality Management: Why, What and How, J. Ross Publishing, Florida.

[105] Sargent, R. G. (1984). A tutorial on verification and validation of simulation models. winter simulation conference.

[106] Seepersad, C.C., Pedersen, K., Emblemsvag, J., Bailey, R., Allen, J.K., and Mistree, F., (2006), "The validation square: How does one verify and validate a design method?"

[107] Sridhar Nerur, RadhaKanta Mahapatra, George Mangalaraj, 2005, Challenges of Migrating to Agile Methodologies, http://miklp.free.fr/p72-nerur.pdf

[108] Stephan Volker, Gabriela Prostean, Andrei Hutanu, 2014, Research of Automotive Change Management and supportive Risk-Management

[109] Thomas, G. & Fernandez, W. (2008), Success in IT projects: A matter of definition?

[110] Turk, D., France, R., Rumpe, B., 2002, Limitations of Agile Software Processes, Agile Alliance.

[111] Wateridge J. 1995, IT projects: a basis for success. Int J Project Manage;13(3):169–72.

[112] Westland, J, 2007, The Project Management Life Cylce, A complete step-by-step methodology for initiating, planning, executing & closing a project succesfully, Kogan Page; London.

[113] White, D. & Fortune, J. 2002, Current practice în project management – an empirical study, International Journal of Project Management, Vol. 20, pp. 1-11.

[114] www.cmcltd.com/sbu/Images/cet_addie.gif. Accesat in data de 27.12.2014.

[115] Y. Yusuf, E. Adeleye, 2002, A comparative study of lean and agile manufacturing with a related survey of current practices în the UK.

[116] YK Malaiya, J Denton, 1999, Requirements volatility and defect density