Georgiana AVRAM (căs. OLARU)
Școlii Doctorale de Economie,
Universitatea Al. I. Cuza,
Iași, România

METODOLOGII DE DEZVOLTARE A PROGRAMELOR INFORMATICE - STUDIU COMPARATIV

Keywords
Metodologii de dezvoltare
Agile
Modelul cascadă
Modelul în V
Modelul spirală
Orientat obiect

JEL classification
L86, C88, M15

Abstract

Mutaţiile din domeniul tehnologiei informaţionale şi al metodelor de abordare a sistemelor s-au reflectat şi în ciclul de viaţă al dezvoltării sistemelor, fie prin schimbarea etapelor acestuia, fie prin modificarea opticii de parcurgere a lor. Spre exemplu, odată cu abordarea orientată-obiect a sistemelor, s-au lansat şi noi modele ale ciclului de viaţă. Prin parcurgerea materialelor de specialitate, se poate constata că numărul fazelor/etapelor unui ciclu de viață variază de la trei (de exemplu analiza, proiectarea, implementarea) la peste douăzeci. Există multe metodologii de dezvoltare a sistemelor, şi fiecare dintre ele este unică în funcție de ordinea și importanța fiecărei faze din ciclul de dezvoltare. Unele metodologii reprezintă standarde formale folosite de agenții guvernamentale, în timp ce altele au fost dezvoltate de firme de consultanţă pentru a fi vândute clienţilor. Multe organizații au metodologiile lor interne care au fost rafinate de-a lungul anilor.

Lucrarea aceasta își propune să facă un scurt inventar al principalelor metodologii de dezvoltarea a programelor informatice, evidențiind diferențele și asemănările dintre ele, ca în final să ofere sugestii cu privirea la oportunitatea folosirii unei metode sau alta în funcție de tipul proiectului.

INTRODUCERE

Sistemele informatice (SI) se caracterizează printr-un ciclu de viaţă care începe cu decizia realizării unui nou SI care să corespundă mai bine noilor cerinţe ale utilizatorilor şi se încheie cu decizia de înlocuire a SI existent cu unul nou, mai performant. Ciclul de viaţă se desfăşoară pe etape în cadrul fiecăreia fiind definite faze şi activităţi specifice.[1] La începuturile dezvoltării programelor informatice, programatorii nu înțelegeau nevoia de a avea o metodologie de dezvoltare bine planificată. După o fază foarte scurtă și simplă de planificare treceau direct la faza de implementare.

METODOLOGII DE DEZVOLTARE A PROGRAMELOR INFORMATICE

Conform definiției de pe Wikipedia, o metodologie de dezvoltare software reprezintă o platformă folosită pentru a structura, planifica și controla procesul dezvoltării unui sistem informatic. Conform autorilor Dennis A.,Haley Wixom B., Tegarden D, în cartea "System Analysis Design UML Version 2.0" , apărută la editura John Wiley & Sons în 2009, o metodologie reprezintă o abordare formalizată a implementării ciclului de viață al dezvoltării unui sistem – o listă de pași și livrabile.

Există multe metodologii de dezvoltare a sistemelor, și fiecare dintre ele este unică în funcție de ordinea și importanța fiecărei faze din ciclul de dezvoltare. Unele metodologii reprezintă standarde formale folosite de agenții guvernamentale, în timp ce altele au fost dezvoltate de firme de consultanță pentru a fi vândute clienților. Multe organizații au metodologiile lor interne care au fost rafinate de-a lungul anilor.

Principalele modele de dezvoltare a sistemelor enumerate de autorul Dumitru Oprea în cartea *Analiza și proiectarea sistemelor informaționale economice*, apărută la Iași la Editura Polirom în 1999, sunt reprezentate de modelul cascadă, modelul incremental, cel evolutiv, în spirală, tridimensional, modelul în V, modelul în X, RAD, Fântană arteziană, Pinball, piramidă, baseball, modelul Agile. În continuare vom descrie câteva dintre ele și caracteristicile lor :

1. *Modelul cascadă (waterfall).*

Fazele propuse de modelul în cascadă sunt

- Realizarea specificațiilor
- Proiectarea aplicației
- Implementarea și integrarea
- Testarea (validarea)
- Instalarea
- Mentenanța

Într-un model cascadă strict, este necesară finalizarea fiecărei faze pentru a trece la următoarea. Revizuirea unei etape poate avea loc doar înainte de a trece la faza următoare și este uneori folosită pentru a valida completitudinea unei faze. Încheierea unei faze este similară unei porți prin care proiectul trebuie să treacă pentru a avansa la faza următoare. Rigiditatea modelului a reprezentat mereu o sursă de critică pentru modelul cascadă.[2]

2. *Modelul spirală.*

Caracteristica cheie a modelului în spirală este reprezentată de prezența managementului riscului în diverse etape ale ciclului de dezvoltare. A fost prezentat în 1988 de Barry Boehm și combină elemente cheie ale modelului cascadă și a metodologiei Rapid Application Development (RAD). Modelul în spirală propune o analiză iterativă a riscului, potrivită mai ales proiectelor mari.

Astfel, modelul spirală poate fi văzut ca un proces care trece printr-un număr de iterații reprezentative pentru urmatoarele activități:[3]

- Determinarea obiectivelor: formularea planurilor pentru a identifica cerințele, pentru a identifica restricțiile de dezvoltare etc
- Analiza riscului: evaluare analitică a programelor selectate pentru a identifica riscurile și metodele pentru a-l elimina.

[1] Stanciu V. – *Proiectarea sistemelor informatice de gestiune*, Ed. Cison, Bucureşti 2000

[2] Hoffer, J., George, J., & Valacich, J. 2006.*Modern systems analysis and design 6th.* Prentice Hall: U.S.A.

[3] Dennis Alan, Barbara Haley Wixom, David Tegarden, *System Analysis Design UML Version 2.0 An Object-Oriented Approach, Third Edition*, John Wiley and Sons, 2009

- Implementarea proiectului și verificarea

Modelul în spirală are și unele aspecte restrictive, după cum urmează:

- Pune un accent deosebit pe analiza riscului și de aceea presupune ca și clienții să accepte acest aspect. Acest fapt presupune atât încredere în programatori cât și dispoziția de a petrece mult timp pentru a rezolva problemele apărute – acesta este unul din motivele pentru care acest model este folosit de obicei pentru proiectele interne de scară largă
- Dacă implementarea analizei riscului afectează într-o mare măsură profitul proiectului, atunci acest model nu ar trebui folosit.
- Pentru ca acest model să fie un succes, programatorii trebuie să caute mereu posibile riscuri și să le analizeze.

3. *Modelul Agile*

Modelul Agile folosește ca și bază modelul iterativ, dar propune un punct de vedere mai deschis și orientat spre oameni decât abordările tradiționale. Procesele Agile încorporează iterația și feedback-ul continuu pentru a rafina succesiv produsul. Metodologia Agile încearcă să captureze dinamica schimbării inerente în dezvoltarea de software și să o folosească în loc să i se opună într-un mediu care se schimbă atât de rapid.[4] Metodele tradiționale presupun ca specificațiile să fie complete înainte de a începe dezvoltarea pe când metodele Agile pleacă de la supoziția că schimbarea nu poate fi evitată și trebuie înglobată în ciclul de dezvoltare.[5]

Există mai multe variații ale proceselor Agile:

- *Extremme Programming (XP)* – fazele sunt îndeplinite în pași extrem de mici în comparație cu procesele mai vechi pe loturi. Prima trecere prin acești pași poate dura o zi sau o săptămâna, în comparație cu luni sau ani – cât durează fiecare pas din modelul cascadă.
- *SCRUM* este cel mai folosit proces Agile - 7th Annual State of Agile Development Survey[6] arată că 72% din respondenți practicau SCRUM sau o variație a acestuia în 2012.
- Metoda dezvoltării dinamice a sistemelor (Dynamic systems development)

4. *Rapid application development (RAD)*

RAD este o metodologie de dezvoltare care folosește o planificare minimală în favoarea realizării de prototipuri. Lipsa unei planificări excesive permite în general scrierea mai rapidă a codului și schimbarea mai ușoară a cerințelor. RAD implică metode ca dezvoltarea iterativă și prototipizarea software-ului și poate fi văzut și ca o fuziune de variate technici structurate, în mod special data-driven information engineering, cu tehnici de prototipizare pentru a accelera dezvoltarea de sisteme software.[7]

5. *Code and fix*

Dezvoltarea "Code and fix" (sau "Cowboy coding"), după cum sugerează și numele, nu reprezintă în mod deliberat o strategie de dezvoltare , ci mai degrabă reprezintă un artefact al naivității și presiunii termenelor limită asupra programatorilor. Fără prea multă proiectare, programatorii încep să scrie cod imediat și la un moment dat începe testarea, urmând ca bugurile găsite să fie rezolvate înainte de punerea în producție.

6. *V-Model*

Modelul în V reprezintă un proces de dezvoltare software care poate fi considerat o extensie a modelului în cascadă. Acesta este criticat de susținătorii modelului Agile deoarece răspunde mai degrabă nevoilor managerilor, contabililor și avocaților decât nevoilor programatorilor și utilizatorilor; este inflexibil și încurajează o viziunea lineară și rigidă a dezvoltării software și nu are abilitatea de a se adapta schimbărilor.

Pe de altă parte, susținătorii acestui model spun că a evoluat de-a lungul timpului și susține flexibilitatea și agilitatea în cadrul procesului de dezvoltare. În plus, este văzut ca o abordare disciplinată ce promovează o

[4] Erickson, J. 2005. *Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research*. Journal of database management: 88-100.

[5] Cao, L., Ramesh, B. 2007. *Agile software development: ad hoc practices or sound principles*. IEEE computer Society: 41-47.

[6] http://www.versionone.com/state-of-agile-survey-results/ , accesat 6 iunie 2013

[7] Whitten, Jeffrey L.; Lonnie D. Bentley, Kevin C. Dittman. (2003). *Systems Analysis and Design Methods*. 6th edition

proiectare meticuloasă, dezvoltarea și documentația necesară pentru a construi produse software stabile. În ultimul timp a fost folosit în mod deosebit în industria dispozitivelor medicale. Modelul în V se pretează și in mediul orientat obiect, deoarece încurajează prototipizarea, realizarea unor structuri superioare și oferă un control puternic asupra sistemului în curs de realizare, prin structurile ierarhice și modulare pe care le promovează, ceea ce îl face utilizabil și în cazul sistemelor complexe.[8]

ADECVAREA METODOLOGIILOR LA TIPURILE DE PROIECT

După cum am văzut, există o paletă largă de metodologii de dezvoltare a proiectelor informatice. Nu există o metodologie care să garanteze succesul proiectului. Alegerea unei metodologii pentru un anumit proiect depinde de o serie de factori, de la mărimea proiectului, criticitatea lui, până la factori ca dinamismul mediului sau cultura organizațională.

În tabelul de mai jos am comparat modul în care folosirea modelele tradiționale sau a celor Agile este impactată de diverși factori ai proiectelor. Am ales această comparație deoarece în ultima perioada curentul Agile a câștigat foarte mulți adepți, pe bună dreptate bineînțeles. Acest lucru nu diminuează totuși avantajele pe care au metodele tradiționale, care reprezintă o alegere inspirată pe anumite tipuri de proiecte.

După cum se poate vedea, nu există o metodă care să se potrivească perfect pe orice tip de proiect. În funcție de mărimea și criticitatea proiectului se va folosi o metodologie sau alta. Deși în ultimii ani curentul Agile a câștigat foarte multi adepți, acesta nu poate fi folosit în proiectele din zona medicală, militară, aero-spațială etc. Aceste industrii implică un grad ridicat de risc, sunt proiecte cu o criticitate deosebită, iar metodele tradiționale sunt mai potrivite pentru a răspunde cerințelor acestor domenii.

Există organizații care au standarde și politici care îi ghidează în alegerea unei metodologii, în timp ce alte organizații au o singură metodologie oficială sau mai multe metodologii oficiale. Fără îndoială că există și organizații care nu au nici o politică în ceea ce privește alegerea metodologiei de dezvoltare a unui proiect. În continuare, vom prezenta o altă serie de criterii propuse pentru a veni în sprijinul celor care căută cea mai potrivită metodologie pentru proiectul lor : claritatea specificațiilor, gradul de cunoaștere a tehnologiilor, complexitatea sistemului, fiabilitatea sistemului, durata sau vizibilitatea planificării.[9]

Claritatea specificațiilor. Când specificațiile pentru un nou sistem sunt neclare devine dificil să le înțelegi sau să le explici în scris. În aceste cazuri sunt recomandate de obicei prototipurile pentru ca utilizatorul să vadă ceea ce poate să facă noul sistem și să-l poată adapta nevoilor lor. Astfel, pentru proiectele cu specificații neclare sunt protrivite metodologiile bazate pe prototipuri – RAD.

Gradul de cunoaștere a tehnologiilor. Atunci când dezvoltarea unui nou proiect se face folosind o tehnologie cu care analiștii și programatorii nu sunt familiari, aplicarea ei trebuie să se facă într-un stadiu incipient al proiectului, pentru a verifica dacă toate funcționalitățile pot fi implementate. Și în acest caz metodologiile bazate pe prototipuri sunt utile, deoarece programatorii vor dezvolta prototipuri pentru funcționalitățile cele mai importante. Și metodologiile bazate pe faze sunt utile în această situație deoarece le oferă programatorilor ocazia să studieze noua tehnologie până la finalul fazei de design.

Complexitatea sistemului. Sistemele complexe necesită o analiză atentă și detaliată a sistemului. Metodele tradiționale gestionează bine sisteme complexe, dar adăugând la ele și implicarea clientului într-o fază incipientă a proiectului, putem evita o serie de probleme care ar putea apărea mai târziu în proiect.

Fiabilitatea sistemului. În cea mai mare parte a proiectelor informatice, fiabilitatea sistemului este foarte importantă. Totuși există diverse stadii de fiabilitate: pentru aparatura medicală spre exemplu, fiabilitatea este critică, în timp ce pentru jocurile video această nu mai are un rol deosebit.

[8] Oprea Dumitru, *Analiza și proiectarea sistemelor informaționale economice*, Iași, Editura Polirom,1999

[9] Dennis Alan, Barbara Haley Wixom, David Tegarden, *System Analysis Design UML Version 2.0 An Object-Oriented Approach, Third Edition*, John Wiley and Sons, 2009

Metodologia RAD permite atât crearea unor specificații detaliate cât și crearea mai multor prototipuri.

Durata proiectului. Pentru proiectele cu o durată mică, cele mai potrivite metodologii sunt RAD și Agile, în timp ce modelul în cascadă reprezintă cea mai puțin inspirată alegere, deoarece nu suportă foarte ușor modificările în planificare.

Vizibilitatea planificării. Una din cele mai mari provocări în dezvoltarea de aplicații informatice este reprezentată de posibilitatea de a determina dacă un proiect este în punctul prevăzut în planificare. În metodele tradiționale este dificil de determinat acest lucru deoarece fazele de design și implementare sunt ultimele din ciclu. Metodologiile Agile și RAD mută deciziile critice în fazele de început ale proiectelor, astfel că managerii de proiect pot ști mai ușor dacă se încadrează în planificare sau nu.

CONCLUZII

Din punctul meu de vedere nici o metodologie nu reprezintă garanția succesului. Cheia reușitei este reprezentată, de fapt, de abilitatea de a alege cea mai potrivită metodologie în funcție de specificul proiectului. Deși în ultima perioadă metodologia Agile este adoptată din ce în ce mai des, sunt de părere că, de foarte multe ori, motivația acestei alegeri nu este susținută de studierea nevoilor proiectului, ci mai degrabă de direcția pieței. Criticile aduse metodelor tradiționale sunt întemeiate, dar nu trebuie să uităm și punctele lor forte, care le-au făcut să fie folosite în majoritatea proiectelor de o criticitate și complexitate ridicată.

Bibliografie

[1] Cao, L., Ramesh, B. 2007. *Agile software development: ad hoc practices or sound principles*. IEEE computer Society: 41-47
[2] Dennis Alan, Barbara Haley Wixom, David Tegarden, *System Analysis Design UML Version 2.0 An Object-Oriented Approach, Third Edition*, John Wiley and Sons, 2009
[3] Erickson, J. 2005. *Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research.* Journal of database management: 88-100
[4] Hoffer, J., George, J., & Valacich, J. 2006.*Modern systems analysis and design 6th*. Prentice Hall: U.S.A.
[5] Oprea Dumitru, *Analiza și proiectarea sistemelor informaționale economice*, Iași, Editura Polirom,1999
[6] Stanciu V. – *Proiectarea sistemelor informatice de gestiune*, Ed. Cison, Bucureşti 2000
[7] Whitten, Jeffrey L.; Lonnie D. Bentley, Kevin C. Dittman. (2003). *Systems Analysis and Design Methods. 6th edition*

Anexe:

Tabel 1.
Impactul factorilor proiectului asupra metodologiilor

Factor	Modele Agile	Modele tradiționale
Mărimea proiectului	Ușor de folosit în proiecte și echipe mici. Dependența de cunoștintele tacite limitează scalabilitatea	Metodele au evoluat pentru a gestiona proiecte și echipe mari. Greu de adaptat pentru proiecte mici
Criticitatea proiectului	Nu este testat pe proiecte cu criticitate ridicata. Lipsa documentației și a prototipurilor ar putea reprezenta o problemă	Metodele au evoluat pentru a gestiona proiecte cu un grad ridicat de criticitate. Greu de adaptat pentru produse cu grad scazut de criticitate
Dinamismul mediului	Proiectarea simplă și adaptarea continuă sunt excelente pentru un mediu dinamic, dar reprezintă o potențială costisitoare sursă de rework pentru mediile foarte stabile	Planurile și proiectarea detaliată reprezintă un punct forte pentru mediile foarte stabile, dar reprezintă o potențială costisitoare sursă de rework pentru mediile dinamice.
Personalul implicat in proiect	Presupune prezența continuă a unei mase critice de experți. E riscantă folosirea personalului care nu este familiarizat cu Agile	Prezența unei mase critice de experți este necesară în faza de definire a proiectului
Cultura organizațională	Are succes în organizațiile cu o cultură care promovează libertatea (succes in haos)	Are succes în organizațiile cu o cultură care promovează reguli și proceduri clare (succes in disciplină)

Tabel nr 2
Criteriile pentru alegerea unei metodologii

Abilitatea de a dezvolta sisteme:	Waterfall	RAD	Agile
Cu specificații neclare	Slabă	Excelentă	Excelentă
Folosind tehnologii noi	Slabă	Excelentă	Slabă
Complexe	Bună	Excelentă	Slabă
Fiabile	Bună	Excelentă	Bună
De durată mică	Slabă	Bună	Excelentă
Cu vizibilitate a planificării	Slabă	Bună	Bună