AGILE vs. Waterfall

- Conf.dr. Cristian KEVORCHIAN
- Facultatea de Matematică şi Informatică
- ck@fmi.unibuc.ro

- In Statele Unite, se cheltuiesc peste 250 miliarde USD pentru dezvoltarea de aplicatii IT(Stadish Smart 2014).
- 31% din proiecte
- Valoarea vizeaza 175,000 proiecte.
- Media costurilor pe proiect :
  - pentru companii mari este 2.322.000 USD
  - pentru companii medii este 1.331.000 USD
  - pentru companii mici 434.000 USD
- O mare parte dintre proiecte esueaza.
- „**Software development projects are in chaos**”

”Performante Negative”

Standish Group Research(2015)

- **19 %** din proiecte vor fi scoase de la finantare inaintea datei de finalizare-ne mai reluindu-se niciodata
- **52 %** remodelate(depasirea timpului, a bugetului, sau functionalitati esuate).
- **29 %** dintre proiecte au fost finalizate la timp si in limitele bugetului
- **42%** din functiile initiale se regasesc la finalul proiectului
- **Studiul s-a bazat pe analiza a 50000 proiecte**

Stadish Group Report

%	‘02	‘04	‘06	‘08	‘10	‘11	‘12	‘13	‘14	‘15
Succes	34	29	35	32	33	29	27	31	28	**29**
Remodelate	51	53	46	44	46	49	56	50	55	**52**
Esec	15	18	19	24	21	22	17	19	17	**19**

Situatia înainte de apariția AGILE Manifesto(Stadish Group)

	Esecuri	Remodelate	Succes
2000	23%	49%	28%
1998	28%	46%	26%
1995	40%	33%	27%
1994	31%	53%	16%

Datele au la baza realizarile a 30000 de proiecte

- Costurile pierderii oportunitatilor sunt greu măsurabile, dar se ridica la mii de miliarde de de dolari.
- Exemplu: aeroportul oraşului Denver pierde 1.1 milioane USD/zi in urma eşecului proiectarii unui produs software pentru managementul bagajelor.
- In 1995 companiile americane şi agenţiile guvernamentale au cheltuit 81 miliarde dolari pentru proiecte de software anulate.
- Aceleaşi organizaţii vor plăti o sumă suplimentară de 59 miliarde pentru proiectele software care vor fi finalizate, dar vor depăşi estimările iniţiale, de timp si bani.

Costul Eseculuі in Dezvoltarea Proiectelor Software

CHAOS 2015 –Factori de succes

A	Sprijinul executivilor	15
B	Maturitate emoțională	15
C	Implicarea utilizatorului	15
D	Optimizare	15
E	Resurse de competențe	10
F	Arhitecturi standard	8
G	Procese Agile	7
H	Utlizarea cu moderatie a instr. PM	6
I	Expertiza PM	5
J	Obiective de business clare	4

Stratificarea CHOS după dimensiunea proiectului

	Complexitate		
	Succes	**Remodelate**	**Eșuate**
Generale(impact global)	**100**	**250**	**400**
Mari	**175**	**325**	**475**
Medii	**250**	**400**	**550**
Moderate	**325**	**475**	**625**
Mici	**400**	**550**	**700**

Chaos Raport-Agile vs Waterfall

Dimensiunea	Metoda	Succes	Remodelate	Esuate
Toate dimensiunile de proiecte	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Proiecte de dimensiuni mari	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Proiecte de dimensiuni medii	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Proiecte de mici dimensiuni	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

Care sunt obstacolele?

- Separarea businss-ului de tehnologie
- Descrierea defectuasa a proceselor de business:
  - Neclaritatea tintelor
  - Lipsa unui management al schimbarii
- Procesele inflexibile la schimbare
- Lipsa comunicarii si realizarea echipei.

Cine dezvolta aplicatii?

- **Scriu software şi au studii de informatică [633]- 48.62**
- **Scriu software şi nu au studii de informatică[637]- 48.92**
- Nu scriu software şi au studii de informatică[22]-1.69
- Nu scriu software şi nu au studii de informatică [10] - 0.77
- **N= 1302**

TESTAREA CODULUI

Da, dezvoltatorii ar trebui să testeze propriul lor cod [303]- 29.53
Da, pentru unele teste, nu pentru altele (de exemplu, teste functionale, de acceptare)[500]- 48.73
Da, dacă nu există nimeni altcineva care poate testa pentru ei[112] 10.92
Nu, toate testele ar trebui să fie planificate, scrise şi executate de testeri specializaţi[58]-5.65
Clienţii testeaza codul[33]- 3.22

Hacking specializat în identificarea bug-urilor

- Salariu 250000 USD/an
- Recompensarea identificării bug-urilor datează din 1995
- De la Microsoft la Tesla au existat programe de recompensare pentru identificarea bug-urilor
- Apple plătește 200000 USD/bug
- Inițiativa ”Hack the Pentagon”

Metodologii de Dezvoltare Software

- Framework utilizat pentru structurarea, planificarea şi controlul procesului de dezvoltare a unui sistem software, care include rezultate specifice predefinite, artefacte care sunt create şi completate de către echipa de proiect pentru a dezvolta sau de a întreţine un sistem software
  - MSF – Microsoft Solution Framework
  - Oracle ADF- Oracle Application Development Framework
  - IBM RUP-Rational Unified Process

- CASE(Computer Aid Software Engineering)

Terence Parr

Associate professor, University of San Francisco
Principles of Software Development

- **Writing software is more an art than an engineering discipline**

Writing software is most similar to writing fiction novels. Writing novels is also an act of creation in an unconstrained and ethereal medium with few well-established construction rules. We know good writing when we see it, but it is hard to teach. Experience writing and feedback from better writers (coders) is the most reliable means of becoming a good writer (coder). Without a well understood process, software will remain more an art than a science. The term "software engineering" is more a goal than how we actually write software.

AGILE IT SURVEY-OCTOBER 2010

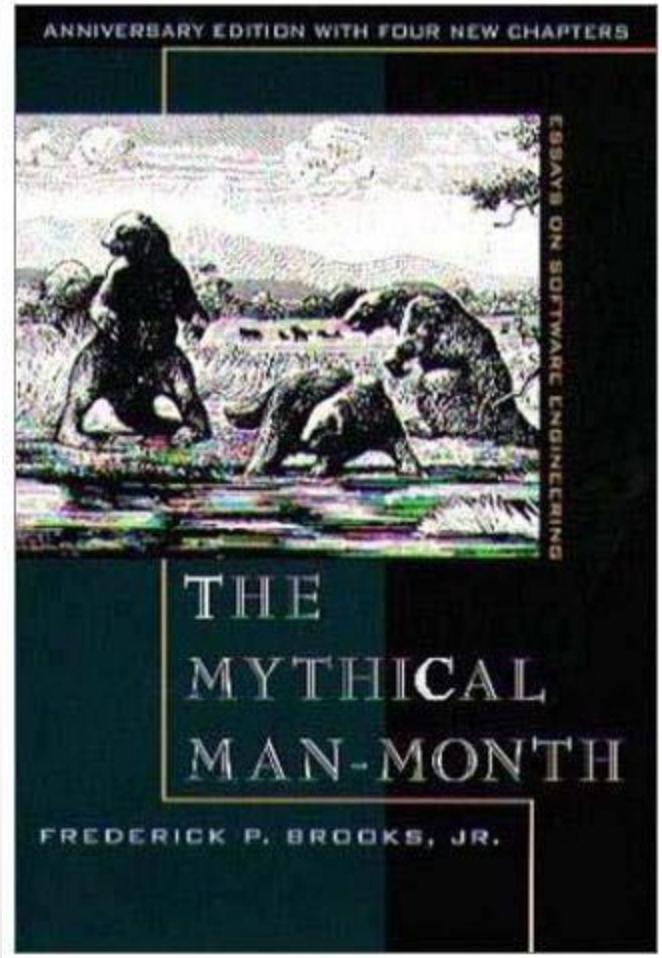
- 88% - volumul proiectelor dezvoltate utilizind metodologia AGILE va creste
- 59% din organizatiile Europene utilizeaza(vor utiliza) o combinatie AGILE – waterfall
- 5% din organizatiile europene intentioneaza utilizarea eminamente AGILE
- 12% intentioneaza sa utilizeze waterfall.
- 15% PPM solution based on industry standards
- 28% PPM solutii desktop bazate pe EXCEL si Microsoft Project

Modelele Waterfall și Sashimi

- Modelul waterfall se bazeaza pe secventialitatea proceselor in care dezvoltarea, asemanator unei cascade trece prin fazele de **Conceptie, Initializare, Analiza, Proiectare**, **Constructie, Testare si Intretinere**.
- Standard DOD-STD-2167,1994 si in 1998 IEEE-12207
- Modelul sashimi, sau “model waterfalls cu feedback” permite suprapunerea fazelor.

THE MITICAL MAN-MONTH

FREDERICK BROOKS



”My rule of thumb is 1/3 of the schedule for design, 1/6 for coding, 1/4 for component testing, and 1/4 for system testing”.

"Conceptual integrity is *the* most important consideration in system design."

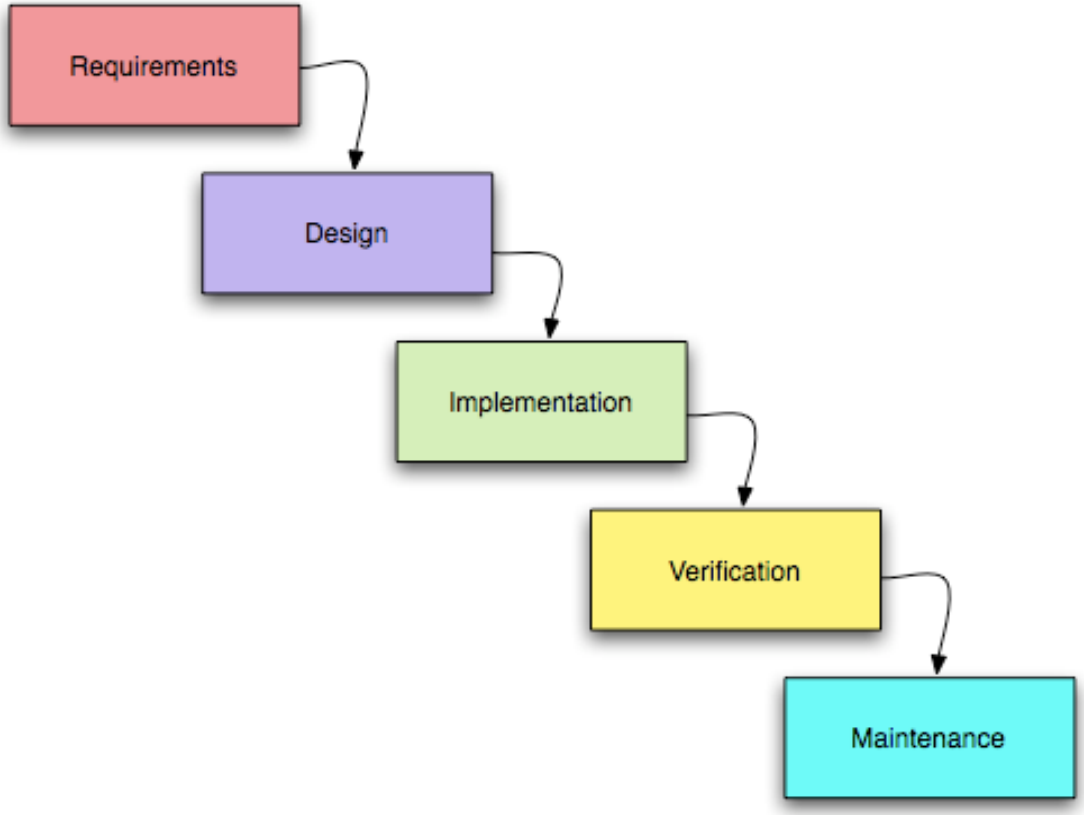
"Separation of architectural effort from implementation is a very powerful way of getting conceptual integration on very large projects."

“A conceptually integrated system is faster to build and to test.”

“Brooks's OS/360 data agrees with Harr's: 0.6-0.8 KLOC/ man-year on operating systems and 2-3 KLOC/man-year on compilers”.

Modelul Waterfalls nemodificat

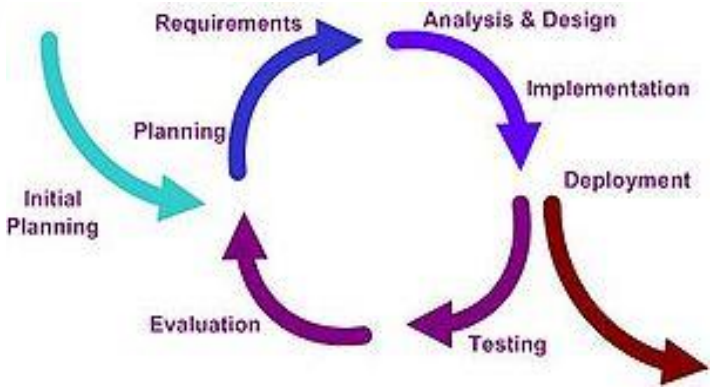
Proiectele pot fi gestionate mai bine atunci când sunt segmentate într-o ierarhie de faze. În proiectele de dezvoltare software:



Esecul Waterfalls

1. Se incearca prevenirea reproiectarii unor componente
2. Din greseli se îmbunătățește dezvoltarea proiectului
3. Reluarea se face in conditiile unei crize acute de timp.
4. Induce risc suplimentar, pentru că amână testarea şi de integrarea până la sfârşitul ciclului de viaţă al proiectului .
5. Recalibrăm scopul şi îl lansăm mai tîrziu.
6. În practică se renunță la testare şi se reproiecteaza

Ciclul de Viata WaterFall vs. Ciclul de Viata Iterativ(I)



Dezvoltarea iterativă şi incrementală este o componenta centrala a unui proces ciclic de dezvoltare software adoptat ca răspuns la punctele slabe ale waterfall. Acesta debuteaza cu o planificare iniţială şi se termină cu instalarea impreuna cu interacţiunile ciclice între ele.

Dezvoltarea iterativa si incrementala sunt parti ale IBM RUP,Microsoft MSF, Oracle ADF

RUP-Rational Unified Process

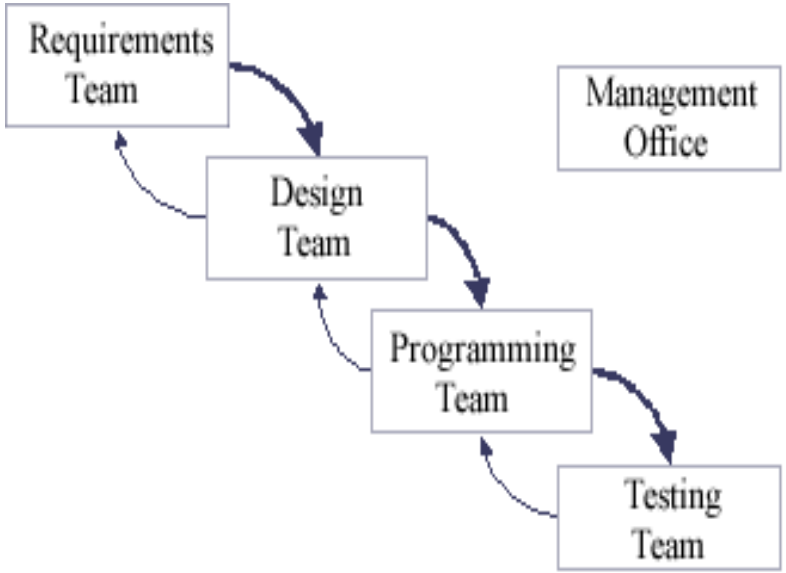
- RUP este un proces configurabil de dezvoltare software care se bazează pe o bogată experiență în utilizarea tehnologiilor bazate pe **obiecte** în dezvoltarea de soluții software în domenii critice aparținînd multor industrii(1998).
- UML – limbaj grafic pentru modelarea analizei și proiectării sistemelor pe baza paradigmei OO(Booch, Jacobson și Rumbaugh).
- RUP este descris ca un “use case driven, arhitecture centric, iterativ and incremental”(Jacobson,1999)
- Use case – descrie o funcționalitate a unui sistem care furnizează o utilitate dat. “Use case model” furnizeaza functionalitatea totala a sistemului.
- Arhitecture centric-blueprint care contine detalii legate de hardware, SO, baze de date, retele, etc
- Iterativ și incremental-imbunatatirea cerintelor genereaza un model in spirala

RUP în WATERFALL

- Dezvoltarea iterativa a componentelor
- Administrarea Cerinţelor
- Utilizarea arhitecturilor bazate pe componente(servicii).
- Modelarea vizuala a arhitecturii software
- Verificarea permanenta a calitatii software-ului.
- Controlul modificarilor in componentele software.
- Alinierea echipei de proiect la arhitectura produsului

Structura echipei WATERFALL

- Cerinte-Requirements Team
- Proiectarea-Design Team
- Dezvoltare-Programming Team
- Testare-Testing Team



“Pentru fiecare problemă complexă există o soluţie simplă, eficientă... şi greşită”-Mencken

- Dezvoltarea iterativa(1957) este la fel de veche ca modelul Waterfall.
  - Smalltalk si LISP au fost iterative
- Avantaje waterfall
  - Avem controlul bugetelor si programului de executie
  - La o privire rapida CMMI(Capability Maturity Model Integration) si PMI sunt waterfalls

Henry Louis Mencken (1880-1956), umorist, jurnalist si critic american

RUP vs. MSN

- Versiunea 7.0 a RUP se defineşte ca un proces în Ingineria Software organizat pe sectiuni şi faze. RUP se bazeaza pe o familie de bune practici de inginerie software, care conţin elementele de bază ale unui proces de dezvoltare (roluri, sarcini, activităţi, artefacte, fluxuri de lucru). RUP dispune de o bibliotecă de concepte din ingineria software. RUP defineşte o modalitate de dezvoltare software care este iterativă și arhitectural-centrica. RUP este în prezent disponibila ca parte a **IBM Rational Method Composer**, care permite adaptarea procesului la nevoile specifice.

- Versiunea actuală 5.0 a MSF se defineşte ca o abordare pentru proiecte care defineşte un set de principii, modele, concepte, ghiduri şi practici deja demonstrate prin Microsoft. Microsoft nu caracterizează MSF drept o metodologie, ci, ca un ghid de bune practici pentru dezvoltarea de aplicații software

Corespondenta MSF-RUP

- Cele doua framework-uri puse fata in fata corespunde la:
  - RUP Inception = MSF Envisioning
  - RUP Elaboration = MSF Planning
  - RUP Construction=MSF developing
  - RUP Transition=MSF Stabilizing and MSF Deploying
- MSF descrie un proces iterativ unde la nivelul fiecarei iteratii trebuie sa existe un livrabil cu toate artefactele necesare.
- Ciclul de dezvoltare RUP are citeva iteratii.La sfirsitul fiecarei iteratii avem un release “intern sau extern”. La sfirsitul ciclului de dezvoltare vom avea un release final.

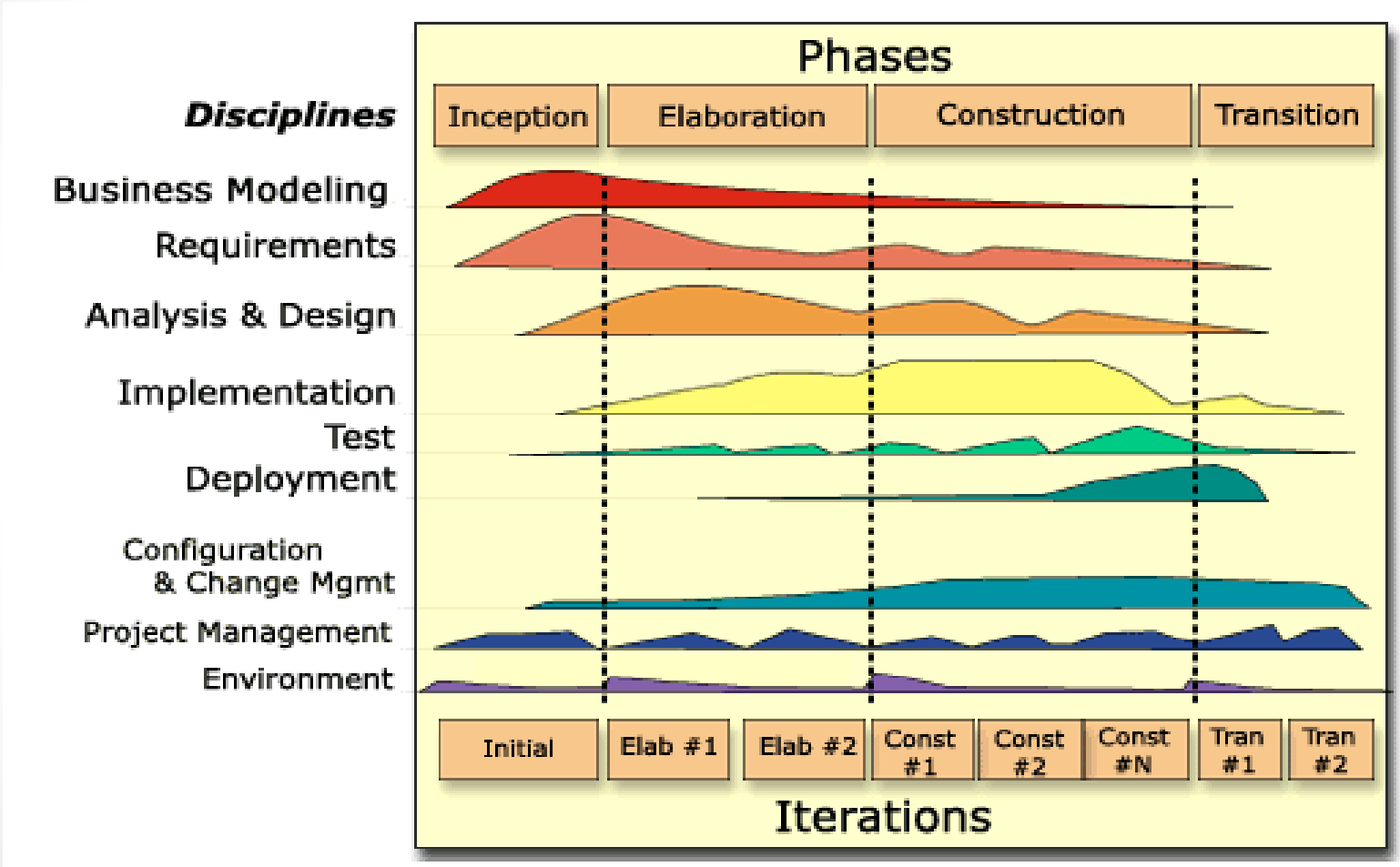
O abordare mai precisa

- Asocierea unei iteratii RUP unei iterattii MSF.
- Asocierea “fazelor” MSF(Viziunea, Planificarea, Dezvoltarea, Stabilizarea, Instalarea) cu “disciplinele RUP”
  - Dimensiunile ciclului de viata RUP
    - **Disciplinele-Modelarea businessului, Cerintele, Analiza si design, Implementare, Testare, Instalare, Config. si Manag. Schimbarii, Project management**
    - Fazele – Initierea, Elaborarea Constructia, Transferul

RUP-Orientat pe Proces

- RUP descrie dezvoltarea unui proces in termenii fazelor care subordoneaza iteratii.
- Fiecare iteratie implica participarea disciplinelor in care gasim roluri, task-uri si artefacte.
- La sfirsitul unei iteratii vom avea o iteratie de delimitare
- La sfirsitul unei faze vom avea o faza de delimitare
- **O familie de faze, iteratii si discipline cuprind un intreg ciclu de viata RUP. La sfirsitul acestui ciclu de viata avem un release complet.**

RUP-Ciclul De Viata

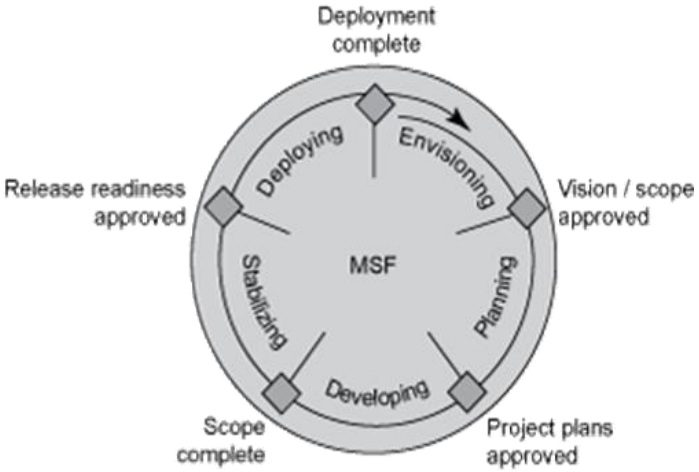


Sursa: http://www.ibm.com/developerworks/rationa

MSF-Orientarea pe proces

- Comparativ cu RUP, MSF este divizat in doua “modele de procese”:
  - **Team Model-activitatile proiectului**
  - **Process Model-secventierea activitatilor proiectului la nivel superior; MSF process model este divizat in faze, fiecare descrie o familie de produse si obiective care trebuie indeplinite**

MSF Team model



MSF Process model

Corespondenta celorlalte elemente

- Artefacte
  - MSF-”livrabile”
  - RUP-”livrabile”, “artefacte”,”rezultate”
- Roluri
  - MSF Team Role : Testing
  - RUP Role:Test Analyst,Test Designer,Test Manager,Tester

- AGILE

Metodologii AGILE

- Metodele de dezvoltare software „expeditivă" au apărut la mijlocul anilor 1990 ca o reacţie la aşa-numitele metode „elaborate", care au fost caracterizate drept puternic reglementate, rigide cum ar fi modelul waterfall de dezvoltare. Propunerea de metode „expeditive”, precum metodele agile, revin la practicile de dezvoltare de la inceputurile istoriei dezvoltarii software.
- Metodele expeditive includ:
  - Scrum (1995)-metodologie iterativ incrementala pentru management de proiect vazuta ca metoda AGILE.
  - Crystal Clear
  - Extreme Programming (1996)
  - Adaptive Software Development
  - Feature Driven Development
  - Dynamic Systems Development (DSDM) (1995).

Acestea sunt acum denumite metodologii AGILE, după Manifestul Agile publicat în 2001

Dezvoltarare AGILE

- Criza software 1960-depasirea timpilor de livrare, a bugetelor conjugat cu nivelul scazut al calitatii solutiilor
  - Solutia #1 Metodele structurate (‘80)
  - Solutia #2 Metodologiile orientate obiect
- Cronicizarea crizei anilor ’60
  - Solutia #3 Imbunatatirea proceselor software(CMM)
  - Solutia #4 Metodologiile AGILE
  - Soulția CroudSourcing(www.viespar.ro)

Principiile AGILE MANIFESTO

- prioritatea constă în a satisface clientul prin livrare rapidă și continuă de componente software funcţionale.
- este bine venită modificarea cerinţelor, chiar târziu în procesul de dezvoltare. Procesele agile valorifică schimbarea pentru a realiza avantaj competitiv clientului
- Livrează componentele software in perioade de timp de la câteva săptămâni până la a câteva luni, cu respectarea timpului minim.
- Analistii de business şi dezvoltatorii trebuie să lucreze, zilnic, împreună pe tot parcursul proiectului
- Realizarea proiectului se bazează pe motivarea membrilor echipei. Mediul de lucru trebuie sa se bazeze pe sprijin și încredere reciprocă în scopul realizarii proiectului.

Principiile AGILE MANIFESTO

- Metoda cea mai eficientă şi eficace de informare în cadrul echipei de dezvoltare este conversaţia „faţă în faţă”.
- Software-ul functional este prima măsura a progresului.
- Procesele agile promovează dezvoltarea durabilă a proiectului. Finanţatorii, dezvoltatorii şi utilizatorii ar trebui să fie capabil de a menţine constant ritmul lucrarilor pe o perioadă nedeterminată.
- Excelenţă tehnică şi design-ul de cea mai bună calitate îmbunătăţeşte agilitatea.
- Simplitatea(**arta de a maximiza cantitatea de muncă neexecutata**) este esenţială.
- Cele mai bune arhitecturi, cerințe şi proiecte sau modele rezulta din auto-organizarea echipelor
- La intervale regulate, echipa analizeaza modul cum să devină mai eficienta, cu calibrarea comportamentului său drept consecință.

Definitia metodologiei AGILE

Metodele de dezvoltare AGILE aplică un timing iterativ dar si o dezvoltare evolutiva, planificare adaptiva impreuna cu o livrare evolutiva a solutiei, care este conjugata cu alte valori si practici care incurajeaza agilitatea. Daca metodele AGILE au un motto atunci acesta este **„Schimbarea”.** Daca metodele AGILE au un punct strategic acesta este manevrabilitatea.

Craig LARMAN

AGILE and ITERATIVE DEVELOPMENT

Ce este diferit in abordările iterative

- Mai multe proiecte mici(**iteratii**) decit unul mare
- Raspunde la întrebarea : “Care este cea mai mica structura functionala care poata fi impachetata si livrata ?” - **Prima Iteratie**
- Integrarea și testarea fiecărei iteratii
- Feedback de la client pentru fiecare iteratie
- Decide cînd poți încheia

Abordarea Iterativă a Proiectului

- **Managementul riscului**-Componentele cu risc se planifică primele
- **Managementul relatiei cu clientul**-lasa clientul sa decida
- **Managemetul deadline-urilor**-transferă ce ai finalizat la data stabilită
- **Planificare adaptivă și evolutivă** – fii pregatit pentru a trece la planuri alternative **(replanificare)** dupa fiecare operatie
- **Livrabile în regim incremental**-livrează “mici bucati” de functionalitate

Beneficiile Periodizării

- “Work expands to fill the time available”- Legea lui Parkinson
- Mai ușor reținem datele eronate decât funcțiile eronate
- Pași mici…putină complexitate multă productivitate
- Forțați deciziile timpurii și impuneți schimbările necesare

Șase Principii ale Dezvoltarii AGILE

- Dacă listele de cerințe ale clientului există, atunci prioritizează-le
- Livrează componente software funcționale ori de câte ori ai ocazia
- Clientul poate beneficia de componentele software ori de cite ori doreste
- Clientul poate adauga, elimina sau reprioritiza cerințele în orice moment al ciclului de viață
- Se păstrează programul stabilit(în condițiile eventualelor schimbări)
- Se poate încheia la orice moment, utilizând ce este proiectat.

Un Proiect Incheiat cu Succes este

- **Un rezultat al lucrului in echipă și al comunicarii**
- **Lucrul in echipa si comunicarea este rezultatul unei balanțe culturale**
- **Balanța culturală** este o rezultantă a „lucrului bine facut”

Q&A