Dezvoltarea agila

Prof. univ. dr. ing. Florica Moldoveanu

Agile Manifesto – cele 4 valori fundamentale

- Metodologiile agile au aparut din necesitatea de a elimina din dezvoltarea de software activitatile care conduc la intarzierea livrarii de software operational.
- Termenul “dezvoltare agila” - derivat din **Agile Manifesto**, publicat in 2001 de un grup din care au facut parte creatorii unor metodologii existente la acea vreme: Scrum, Extreme Programming (XP), Dynamic Systems Development Method (DSDM) si Crystal.
- Agile Manifesto a stabilit un set de valori și principii generale comune metodologiilor agile existente la momentul respectiv. Sunt definite **4 valori fundamentale** care permit echipelor de dezvoltare software sa fie performante:
  - **Indivizi si interactiuni** în loc de procese si instrumente
  - **Software functional** în loc de documentatie cuprinzatoare
  - **Colaborarea cu clientul** în loc de negociere contractuala
  - **Raspunsul la schimbari** în loc de urmărirea unui plan

Agile Manifesto – cele 12 principii (1)

1. Prioritatea maxima o are satisfacerea clientului prin furnizarea rapida si continua de software functional.
2. Schimbarile in cerinte sunt binevenite, chiar si tarziu in dezvoltare, in avantajul clientului.
3. Livrarea frecventa de software functional, la cateva saptamani sau luni, preferabil la intervale mai scurte.
4. Dezvoltatorii si expertii din domeniul aplicatiei trebuie sa lucreze impreuna zilnic pe toata durata proiectului.
5. Construieste proiecte în jurul unor persoane motivate. Dă-le mediul și sprijinul de care au nevoie, și ai încredere în ei pentru a obtine rezultatul.

   Echipele agile sunt alcatuite din oameni care doresc sa lucreze impreuna colaborativ si sa invete unul de la altul. Oamenii sunt principalul factor de succes in dezvoltarea de software.

6. Cea mai eficace metoda de transmitere a informatiilor este comunicarea directa in echipa.
7. Software-ul functional este principala masura a progresului.

Agile Manifesto – cele 12 principii (2)

8. Procesele agile promoveaza dezvoltarea durabila. Sponsorii, dezvoltatorii si utilizatorii trebuie sa fie capabili sa mentina un ritm constant pe timp nelimitat.

   Pentru a obtine rezultate de calitate, oamenii nu trebuie fortati sa munceasca 12 ore/zi in permanenta.

9. Atenția continua la excelenta tehnica si o buna proiectare maresc agilitatea.

   Practici: refactoring - mentinerea unui cod de calitate si test-driven development - software-ul este functional in permanenta. Se folosesc ghiduri de codare si uneori ghiduri de modelare.

10. Simplitatea – arta de a maximiza cantitatea de munca nefacuta – este esentiala.

    - Focalizare pe ceea ce trebuie livrat, evitandu-se munca inutila, timp consumat inutil, pentru a maximiza ROI

11. Cele mai bune arhitecturi, cerinte si proiectari rezulta din echipe care se auto-organizeaza.

    Agile Model Driven Development (AMDD) și Test Driven Development (TDD) sunt abordările principale din cadrul comunității agile, pentru producerea de arhitecturi eficiente, cerințe, și modele de calitate.

12. La intervale regulate echipa analizeaza cum poate sa devina mai eficienta apoi isi ajusteaza corespunzator comportamentul.

Metodologii de dezvoltare agilă

Metodologii: Extreme programming (XP), Scrum, Crystal, Dynamic Systems Development Method (DSDM), Feature Driven Development (FDD), Behavior Driven Development (BDD), Kanban, s.a.

PRINCIPII GENERALE

Implicarea clientului	Clientul trebuie să fie puternic implicat în procesul de dezvoltare. Rolul lui este de a furniza şi prioritiza noi cerințe şi de a evalua iterațiile dezvoltării.
Livrarea incrementală	Produsul software este dezvoltat în incremente, clientul specificand cerințele care trebuie sa fie implementate în fiecare increment.
Oameni, nu procese	Abilitațile echipei de dezvoltare trebuie sa fie recunoscute şi exploatate. Membrii echipei trebuie lasați să-şi dezvolte propriul stil de lucru, fără să li se impuna procese prescriptive.
Inglobează schimbarea	Se asteaptă schimbarea cerințelor, de aceea sistemul trebuie proiectat pentru a fi uşor adaptat la schimbari.
Menține simplitatea	Focalizare pe simplitate atât în produsul software dezvoltat cât şi în procesul de dezvoltare. Elimină complexitatea oricând este posibil.

Extreme programming (XP)

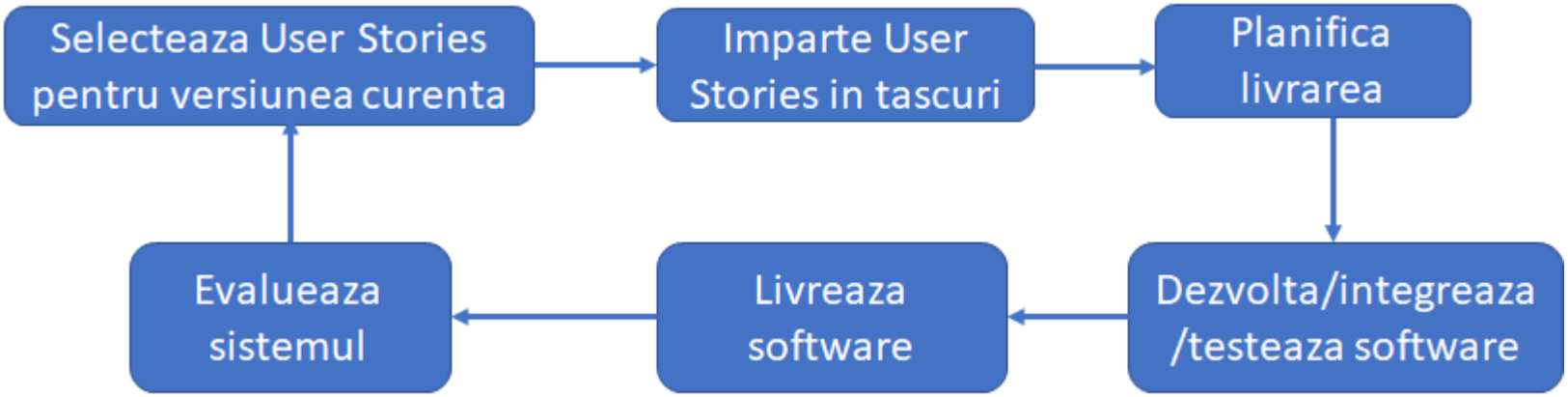
- Aparuta la sfarsitul anilor ’90
- **Abordare “extremă” în dezvoltarea iterativă:**
  - Versiuni noi pot fi construite de mai multe ori pe zi
  - Incrementele sunt livrate clientilor la intervale de 2 saptamani
  - Pentru fiecare versiune executabila trebuie rulate toate testele si versiunea este acceptata numai daca testele au trecut cu succes.

Practici cheie

- “User Stories” pentru specificare
- Test driven development
- Pair programming

XP – User stories

- In XP, **un reprezentant al utilizatorilor finali/clientului face parte din echipa de dezvoltare** si este **responsabil cu cerintele**.
- Cerintele utilizator sunt exprimate sub forma de **scenarii, numite "user stories".**
- User story trebuie sa fie o descriere scurta, fiind limitata la o pagina de hartie de dimensiuni mici (de obicei 3x5 inches – 1 inch =2.54 cm), impiedicand astfel cresterea descrierii.
- Reprezinta o modalitate simpla si rapida de a capta cerintele clientului.
- Pornind de la scenarii dezvoltatorii definesc tascuri de implementat.
- Pe baza tascurilor se fac estimarile de cost si timp.
- **Clientul alege cerintele de implementat in urmatorul livrabil pe baza prioritatii lor si a estimarilor de timp**.



User story

Kate is a doctor who wishes to prescribe medication for a patient attending a clinic. The patient record is already displayed on her computer so she clicks on the medication field and can select 'current medication', 'new medication' or 'formulary'.

If she selects 'current medication', the system asks her to check the dose. If she wants to change the dose, she enters the dose and then confirms the prescription.

If she chooses 'new medication', the system assumes that she knows which medication to prescribe. She types the first few letters of the drug name. The system displays a list of possible drugs starting with these letters. She chooses the required medication and the system responds by asking her to check that the medication selected is correct. She enters the dose and then confirms the prescription.

If she chooses 'formulary', the system displays a search box for the approved formulary. She can then search for the drug required. She selects a drug and is asked to check that the medication is correct. She enters the dose and then confirms the prescription.

The system always checks that the dose is within the approved range. If it isn't, Kate is asked to change the dose.

After Kate has confirmed the prescription, it will be displayed for checking. She either clicks 'OK' or 'Change'. If she clicks 'OK', the prescription is recorded on the audit database. If she clicks on 'Change', she reenters the 'Prescribing medication' process.

http://iansommerville.com/software-engineering-book/

XP – Pair programming

- Dezvoltatorii lucreaza in perechi, creând codul impreuna.
- Este o revizie informala a codului – fiecare linie de cod este vazuta de 2 persoane
- Perechile sunt create dinamic, astfel încat toti membrii echipei lucreaza împreuna pe parcursul dezvoltarii

  → ajuta la cunoasterea codului de toata echipa

  → toata echipa este responsabilă pentru intregul cod (proprietate colectiva)
- Incurajeaza refactorizarea
- Dezvoltarea în echipă reduce riscul proiectului atunci când un membru paraseşte echipa.
- Dezvoltarea in perechi nu este ineficienta: exista dovezi care sugereaza ca este mai eficienta daca cei 2 ar lucra separat.

XP şi principiile agile

- **Dezvoltarea incrementala**: incremente mici livrate frecvent
- **Implicarea clientului**: angajarea permanenta a clientului in echipa.
- **Oameni nu procese**: pair programming, proprietate colectiva, evitarea programului de lucru prelungit.
- **Schimbarile** acceptate prin livrari regulate.
- **Mentinerea simplitatii** prin refactorizarea continua a codului.

XP - limitari

- Fara teste/proceduri de acceptare specificate de client “user stories” sunt supuse interpretarii, ceea ce le face dificil de utilizat ca baza contractuala.
- Solicita contactul permanent cu clientul pe tot parcursul proiectului, dificil in unele cazuri, sau timp suplimentar consumat.
- Dificil de scalat la proiecte mari.
- Presupune dezvoltatori competenti.
- XP este greu de integrat cu practicile de management din cele mai multe companii.
- Desi in dezvoltarea agila se folosesc practici din XP, metoda asa cum a fost definita initial nu este larg folosita.

Scrum

Conform definitiei creatorilor săi, Scrum este un framework care permite abordarea de probleme complexe, oferind în același timp productiv și creativ produse de cea mai înaltă valoare posibilă [1].

Este un framework de dezvoltare agila cu urmatoarele caracteristici esentiale:

❑ Echipa Scrum

- Se auto-organizeaza: nu exista un conducator al echipei care decide ce va face fiecare membru al echipei si cum se rezolva o anumita problema; acestea sunt hotarate de echipă ca un intreg.
- Este multifunctionala: este necesar ca fiecare membru al echipei să fie capabil sa trateze o problema/sarcina de la idee la implementare.
- Este sprijinită de 2 roluri specifice:

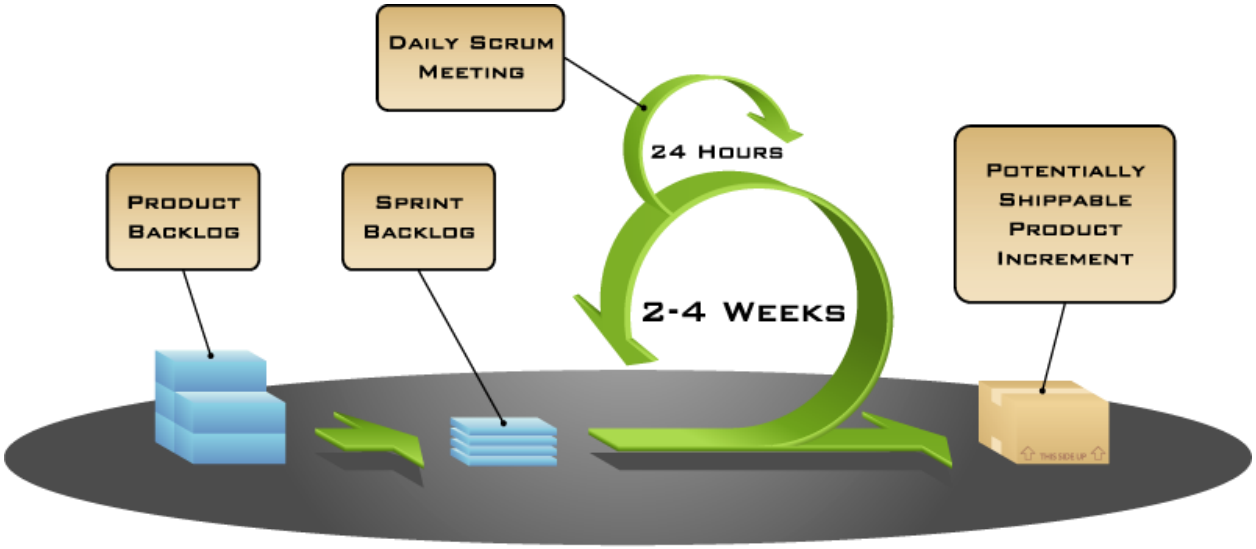
- **ScrumMaster**, care poate fi considerat un antrenor al echipei, ajutand-o sa foloseasca procesul Scrum pentru a performa la nivelul cel mai inalt.
- **Product Owner**, care reprezinta clientul/utilizatorii finali/partile interesate in proiect şi ghideaza echipa pentru construirea produsului aşteptat de acestia.

Caracteristici Scrum (2)

❑ Procesul de dezvoltare

- **Product owner** creaza lista de cerinte prioritizata, numita "**product backlog**"
- Dezvoltarea este organizata in iteratii scurte, de 2-4 saptamani, numite "sprint-uri".
- In cadrul unui sprint este proiectat, codat și testat un increment al produsului final, potential livrabil.
- In fiecare sprint sunt implementate o parte dintre cerintele din "product backlog", selectate de echipă, care formeaza "**sprint backlog**".
- In timpul unui sprint echipa se intalneste zilnic pentru a evalua progresul său (**Daily Scrum**)
- Nu se accepta schimbari in timpul unui sprint, de aceea durata sprint-urilor este planificata de echipă astfel încat sa nu fie necesare modificari in timpul sprint-urilor.
- Sprintul se termina cu un "**sprint review**" si o "**retrospectiva**" a sprint-ului

Procesul Scrum (2)



Sursa: [2]

Scrum framework

Roluri

- Product owner
- ScrumMaster
- Echipa de dezvoltare

Activităţi

- Planificare sprint
- Daily Scrum
- Revizia sprint-ului
- Retrospectiva sprint-ului
- Rafinare product backlog

Artefacte

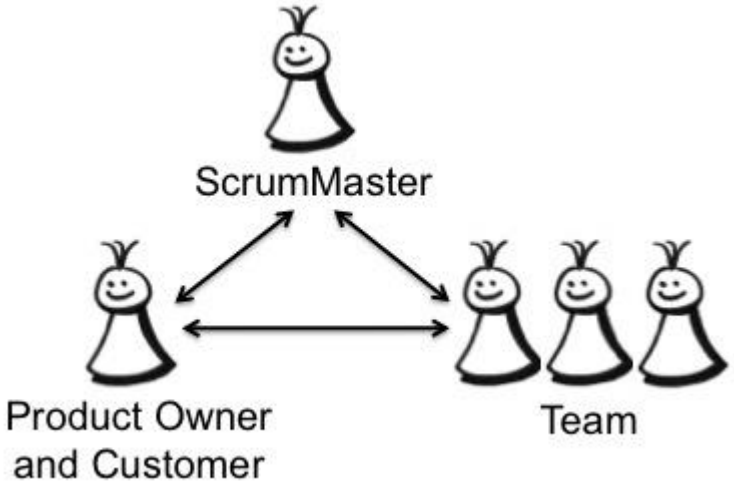
- Product backlog
- Sprint backlog
- Burndown charts
- Versiuni executabile ale produsului

Roluri - Product owner

- Este membrul echipei Scrum care **are sarcina de a maximiza valoarea muncii echipei**:
  - este responsabil pentru profitabilitatea produsului (ROI – Return On Investment)
- Detine viziunea asupra produsului, colaboreaza strans cu partile interesate in proiect (utilizatori, clienti), cultiva şi alimenteaza o comunitate in jurul produsului.
- **Faciliteaza comunicarea dintre echipă şi partile interesate, asigura construirea produsului potrivit (asteptat de clienti, utilizatori).**
- Decide continutul listei de cerinte (product backlog) şi ordoneaza articolele sale după prioritate astfel incat sa se obtina valoarea maxima.
- Decide când va fi lansat produsul și ce funcționalitați va avea.
- Modifică lista de cerințe înaintea fiecarei iterații, după cum e necesar.
- Acceptă sau respinge ceea ce s-a produs.
- Decide impreuna cu echipa care articole din lista de cerințe vor trebui considerate pentru sprintul curent.

Roluri - ScrumMaster

- **Antreneaza echipa**: ajuta intreaga echipa sa obtina performanțe mai bune, sa aplice corect metodologia Scrum, sa aplice practici tehnice care o ajuta sa finalizeze sarcinile fiecarui sprint.
- **Responsabil pentru aplicarea valorilor și practicilor din Scrum**.
- Incurajeaza auto-organizarea echipei.
- Îndepărtează impedimentele.
- Se asigură că echipa este complet funcțională și productivă.
- Încurajează cooperarea strânsă între toate rolurile.
- Protejează echipa de interferențele externe.



Roluri – Echipa

- De obicei 5-9 persoane.
- **Multi-funcțională**:
  - Nu include roluri traditionale ca programator, tester, arhitect
  - Toti din echipa lucreaza împreuna pentru realizarea sarcinilor sprint-ului, pe care si le-au asumat in mod colectiv
- Membrii trebuie sa fie disponibili echipei si proiectului tot timpul.
- **Se auto-organizează**:
  - Product owner creaza o lista ordonata cu ceea ce este necesar sa fie facut
  - Membrii echipei evalueaza cât de mult se poate face într-un singur sprint şi se auto-organizeaza, hotărând ei singuri cine ce face astfel încât la sfarsitul sprintului să se obtina un nou increment al produsului.
- Membrii echipei ar trebui schimbați doar între sprinturi.

Artefacte - Product backlog

- O lista ordonata de idei si cerințe pentru produs, care pot proveni de la product owner, membrii echipei sau partile interesate.
- Fiecare articol din lista este insotit de o estimare a efortului de implementare in numar de ore.
- Lista este ordonata (prioritizata) de către product owner, astfel incat valoarea muncii echipei sa fie maximizata.
- Lista este reordonata la începutul fiecărui sprint.
- De obicei, la inceputul proiectului lista este scurta şi vagă, devine apoi din ce in ce mai lunga şi mai bine definita.
- **Product owner este responsabil pentru crearea şi rafinarea listei dar echipa îl ajuta în producerea si actualizarea listei.**

Un exemplu de product backlog

Articol in product backlog	Estimare
Permite unui oaspete să facă o rezervare	3
În calitate de oaspete, vreau să anulez o rezervare.	5
În calitate de oaspete, vreau să schimb datele rezervării.	3
Ca angajat al hotelului, pot să execut rapoartele RevPAR (revenue-per-available-room)	8
Îmbunătățește gestionarea excepțiilor	8
…	30

Sursa [2]

Activitati - Planificarea sprint-ului

- Are loc in cadrul unei intalniri cu durata limitata la care participa intreaga echipa.
- **Product owner:**

  - prezinta ce anume trebuie facut, utilizand articole din product backlog

  - raspunde la întrebarile echipei pentru a lamuri neînțelegerile asupra articolelor din product backlog

- **Echipa:**

  - decide cate articole sa ia din product backlog şi cum sa le realizeze

  - identifica taskurile şi estimeaza durata pentru fiecare (1-16 ore)

  - creaza sprint backlog-ul

- Durata recomandata pentru o intalnire de planificare sprint este de 2 ore (sau mai putin) x numarul de saptamani ale sprint-ului.
- Succesul intalnirii depinde de calitatea product backlog-ului, de aceea este importanta rafinarea sa.
- Adesea se defineste un scop al sprintului, care ajuta echipa sa se focalizeze pe realizarea sa.
- Se discuta design-ul de nivel înalt.

Artefacte - sprint backlog

- Este lista articolelor din product backlog detaliate, alese pentru implementare in sprintul curent, impreuna cu taskurile planuite de echipa pentru realizarea lor si estimarea efortului.
- Odata stabilit sprint backlog, echipa incepe sa lucreze la noul increment al produsului.

Exemplu de sprint backlog

Task -uri	L	Ma	Mi	J	V
Impl. interf. cu utilizatorul	8	4	8		
Impl. middle tier-ul	16	12	10	4	
Testează middle tier-ul	8	16	16	11	8
Scrie help-ul online	12				
Impl. clasa foo	8	8	8	8	8
Adaugă logarea erorilor			8	4	

Sursa [2]

Meeting-ul SCRUM zilnic

- Are loc la aceeasi ora si in acelasi loc.
- Dureaza maxim 15 minute. Se stă în picioare.
- Nu se rezolvă probleme
  - Este invitata toată lumea
  - Doar membrii echipei, ScrumMaster-ul si Product owner-ul pot vorbi
- Fiecare membru al echipei raspunde la 3 intrebari:
  - Ce ai facut ieri?
  - Ce vei face azi?
  - Exista ceva care te împiedică?
- Acestea NU reprezinta un raport pentru ScrumMaster, sunt angajamente in faţa unor egali.
- In functie de ceea ce s-a discutat la intalnirea zilnica echipa se poate reorganiza, pentru indeplinirea scopului sprintului si finalizarea incrementului.

Activitati - Sprint review

- Intalnire la finalul sprintului, cu durata limitata:
  recomandabil, o ora x nr. de spatamani ale sprintului.
- Informal
  - De regulă, 2 ore pentru pregătire
  - **Fără slide-uri**
- **Participanti**: product owner, echipa, ScrumMaster, conducerea companiei, clienti, dezvoltatori din alte proiecte.
- **Echipa prezintă ce a realizat în timpul sprintului**
- De obicei functionalitatile implementate in timpul sprint-ului sunt demonstrate prin scurte **aplicatii demo.**
- Se discuta ce au observat membrii echipei in timpul sprintului si eventual idei noi
- Se actualizeaza product backlog cu cerintele ramase

Activitati - Retrospectiva sprint-ului

- Intalnire scurta (maxim 3 ore), dupa Sprint review.
- Se discuta despre “ce merge și ce nu merge”.
- **Se identifica potentiale imbunatatiri si se planifica imbunatatirea procesului**.
- Participa:
  - toti membrii echipei
  - eventual clienți și alte persoane
- Scrum master trebuie sa asigure ca retrospectiva are loc si participanții înțeleg scopul ei.
- Incurajeaza membrii echipei sa participle la retrospective.

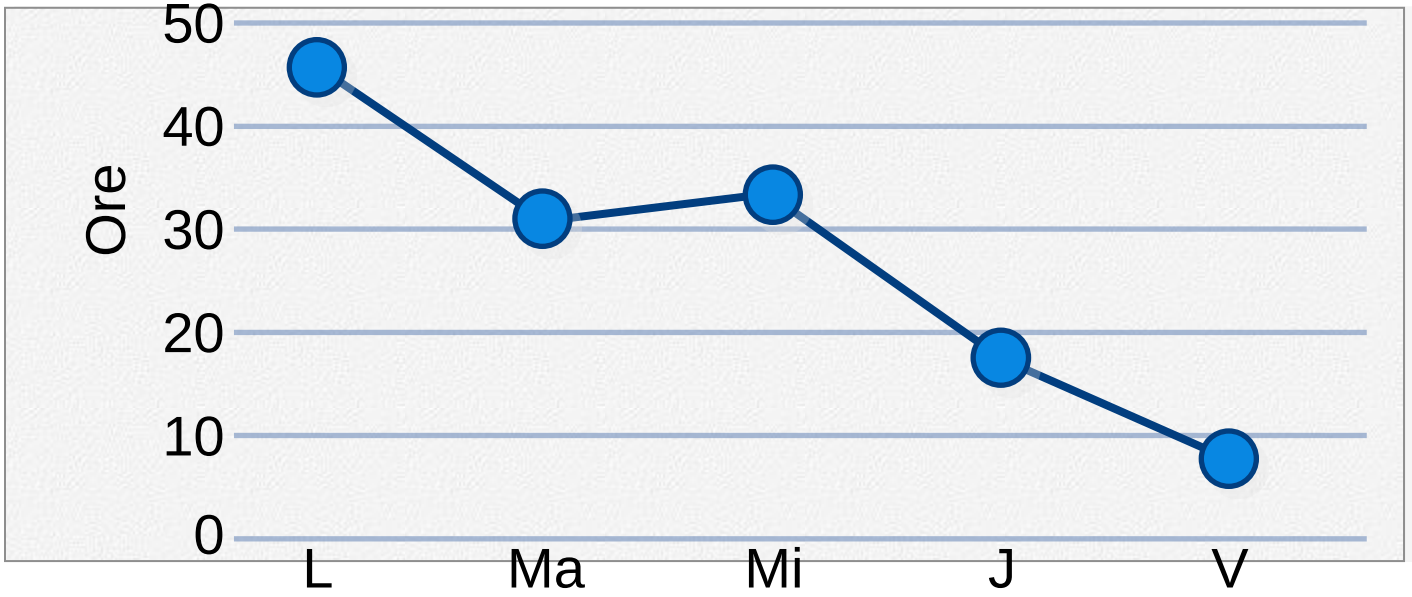
Gestionarea sprint backlog-ului

- Fiecare persoană își alege ce va lucra după propria dorință.
  - Taskurile nu sunt niciodată asignate de altcineva
- **Estimările din sprint backlog sunt actualizate zilnic.**
- Orice membru al echipei poate sa adauge, sa șteargă sau sa modifice articole din sprint backlog.
- Taskurile din cadrul sprint-ului sunt descoperite în mod natural.
- Durata lor este ajustata pe parcursul sprintului.
- **Se actualizează volumul de muncă rămas pe măsură ce se obțin mai multe informații.**

Artefacte - Sprint burndown

Task-uri	L	Ma	Mi	J	V
Impl. interf. cu utilizatorul	8	4	8		
Impl. middle tier-ul	16	12	10	7	
Testează middle tier-ul	8	16	16	11	8
Scrie help-ul online	12				

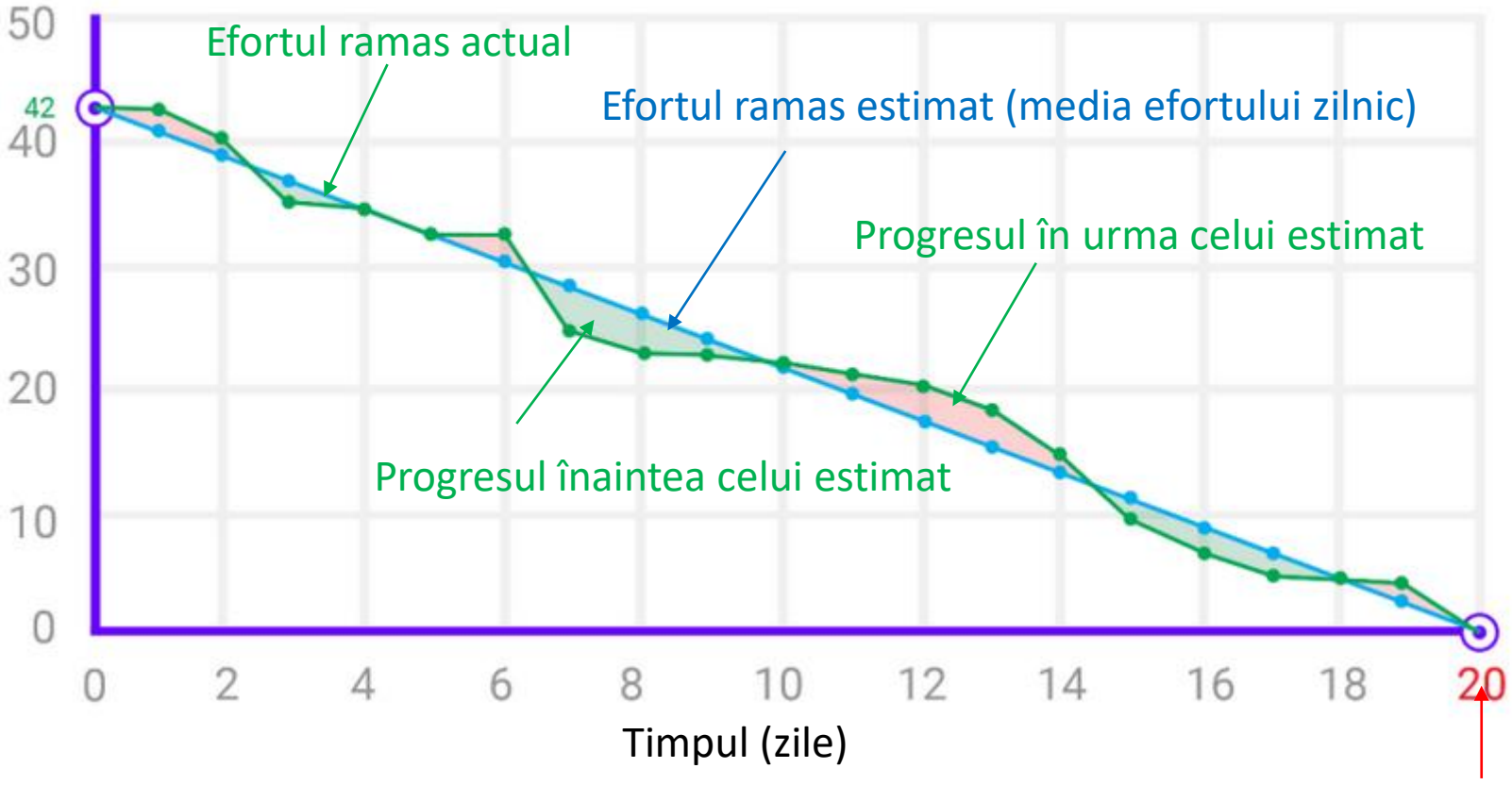
Diagrama sprint burndown: efortul ramas (ore) pana la sfarsitul sprintului



Sursa [2]

Diagrama sprint burndown

Efortul (ore/zile) ramas pana la terminarea sprintului/proiectului



Inaintea/în urma planificarii → ajustarea estimarilor

Rafinarea product backlog-ului

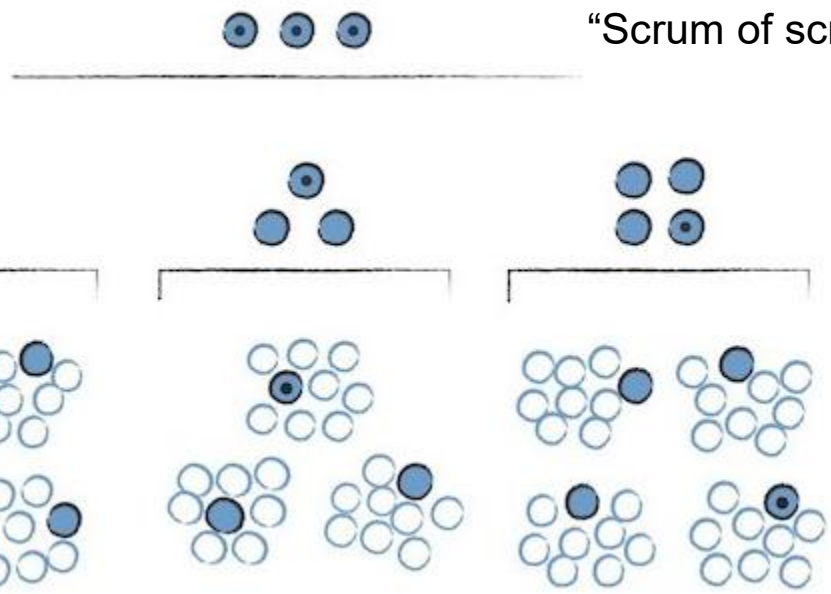
- **Activitate care se desfasoara pe toata durata unui proiect Scrum.**
- Este responsabilitatea intregii echipe.
- Consta din:

- Adaugarea de articole noi sau care au devenit mai importante
- Eliminarea articolelor care nu mai sunt importante
- Ordonarea articolelor
- Impartirea articolelor mari in articole mai mici, astfel incat sa poata fi implementate pe durata unui singur sprint
- Gruparea mai multor articole intr-unul singur
- Identificarea articolelor care ar trebui sa fie selectate in urmatorul sprint

Scalabilitate

- Echipa Scrum tipică e formată din 7 ± 2 persoane
  - Scalabilitatea se obține prin echipe formate la rândul lor din echipe: **scrums of scrums**
- Factori care pot influența scalarea
  - Tipul şi complexitatea aplicației
  - Numarul de persoane estimat pentru dezvoltare
  - Împrăștierea echipei
  - Durata proiectului
- Scrum-ul a fost folosit in mai multe proiecte la care au participat peste 500 persoane
- **Meeting-ul "Scrum of scrums"**
  - Are rolul de a coordona activitatea unui grup de echipe
  - Fiecare echipa din grup, la finalul intalnirii zilnice, desemneaza o persoana care va participa la meeting-ul "Scrum of scrums".
  - Meeting-ul "Scrum of scrums" este analog meeting-ului Scrum zilnic, dar nu are loc in mod necesar in fiecare zi; in multe organizatii are loc de 2-3 ori pe saptamana.

Scrums of scrums



Sursa: [4]

- Meeting-ul “Scrum of scrums” se desfasoara la fel ca o intalnire Scrum zilnica, fiecare participant raportand ce s-a facut, ce se va face si impedimentele, in numele echipei pe care o reprezinta.
- Impedimentele sunt focalizate pe problemele de coordonare intre echipe.
- Rezolvarea impedimentelor poate consta in: acordul asupra interfețelor dintre echipe, negocierea asupra responsabilitatilor, etc.

Metodologia Scrum a fost folosita de:

- Microsoft
- Yahoo
- Google
- Electronic Arts
- High Moon Studios
- Lockheed Martin
- Philips
- Siemens
- Nokia
- Capital One
- BBC
- Intuit
- Intuit
- Nielsen Media
- First American Real Estate
- BMC Software
- Ipswitch
- John Deere
- Lexis Nexis
- Sabre
- Salesforce.com
- Time Warner
- Turner Broadcasting
- Oce

Sursa [2]

Scrum a fost folosita pentru:

- Software comercial
- Aplicații in-house
- Aplicații la comandă
- Proiecte cu preț fix
- Aplicații financiare
- Aplicații certificate ISO 9001
- Sisteme încorporate (embedded)
- Sisteme cu cerințe de disponibilitate 99.999%, 24x7
- Programul Joint Strike Fighter

- Dezvoltarea de jocuri video
- Sisteme aprobate de FDA, life-critical
- Software de control al sateliților
- Site-uri web
- Software pentru dispozitive mobile
- Aplicații pentru rețele de calculatoare
- Unele din cele mai mari aplicații in uz

Sursa [2]

Mai multe informații

1. Ken Schwaber & Jeff Sutherland, The Scrum Guide, https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf#zoom=100
2. Mike Cohn, "O introducere in Scrum": www.mountaingoatsoftware.com/agile/scrum/a-reusable-scrum-presentation
3. www.mountaingoatsoftware.com/scrum
4. https://www.mountaingoatsoftware.com/agile/scrum/team
5. www.scrumalliance.org
6. www.controlchaos.com
7. Craig Larman, *Agile and Iterative Development: A Manager's Guide,* 2004, Pearson Eucation.
8. Mike Cohn, *Agile Estimating and Planning*, 2006, Prentice Hall.
9. Ken Schwaber, *Agile Project Management with Scrum,* 2004, Microsoft Press.