Agile Project Management – SCRUM

1. Definiții

Sprintul – perioada când echipa Scrum lucrează la niște taskuri specifice. Pentru echipele noastre, iterația derulează timp de 22 de zile lucrătoare. Un sprint Include faze de dezvoltare software tradiționale, precum colectarea cerințelor, evaluarea, proiectarea soluției, evoluția, livrare. Echipa de dezvoltare își concentrează eforturile pentru a genera o nouă unitate funcțională a produsului la sfârșitul unui sprint. Cu ajutorul lor, sistemul se adaptează mai bine la schimbare. Sprint-urile au următoarea structură:

- Planificarea Sprintului,
- Întâlnirile Zilnice (Scrums),
- Dezvoltarea produsului,
- Revizuirea Sprintului,
- Retrospectiva.

Backlog de proiect/produs - set de sarcini nerezolvate: descrieri ale funcționalităților și serviciilor dorite (în dependență de ce este nevoie pentru atingerea scopului); poate fi schimbat de oricine. Sarcinile au diverse priorități care le sunt atribuite de către Product Owner, în funcție de valoarea funcțională a lor (care este stabilită de Product Owner) și de efortul necesar pentru dezvoltarea acestora (stabilit de Echipa de Dezvoltare cu Scrum Master/Team Leader). Backlog-ul este actualizat constant prin adăugarea, schimbarea, specificarea, eliminarea și stabilirea priorităților privind funcționalitățile incluse. De asemenea pot face parte din produsul nerezolvat implementarea anumitor funcții, remedierea erorilor, eliminarea defectelor, îmbunătățirea diverselor componente. Printre cei care pot participa la construirea acestui produs se numără clienții, echipa de dezvoltare, echipele de marketing și vânzări. Product Owner-ul este responsabil de gestionarea produsului nefinisat.

Sprint Backlog - este un document detaliat, bazat pe elementele produsului nerezolvat, care va fi abordat în cursul următorului sprint; conține informații despre modul în care Echipa va implementa cerințele stabilite. Sarcinile sunt împărțite în ore, pentru ca nici o activitate să nu dureze mai mult de 8 ore (dacă o sarcină poate dura mai mult, aceasta trebuie împărțită în sarcini mai mici). Sarcinile nu sunt atribuite angajaților - ei sunt liberi să-și aleagă sarcinile dorite. În cadrul ședinței de planificare a sprint-ului, Scrum Master-ul împreună cu Product Owner-ul și Echipa de dezvoltare determină ce elemente vor face parte din sprintul nerezolvat în baza priorităților și obiectivelor stabilite pentru acest sprint. Backlog-ul sprintului este fix până la finalizarea acestuia. Când toate sarcinile sprintului curent sunt îndeplinite, o nouă iterație a sistemului este finalizată și poate fi rezumată într-o Notă de lansare.

Sprint Burndown Chart - este un grafic care prezintă timpul și munca necesare până la finalizarea produsului. Acesta este actualizat zilnic și ajută la estimarea când e data de lansare la care produsul va fi gata.

Estimarea efortului - este un proces iterativ în cadrul căreia atenția este concentrată pe estimarea cât mai precisă a efortului depus pentru a gestiona o sarcină nerezolvată,

Ședința de planificare a sprintului - este o ședință derulată în două etape, organizată de Scrum Master. Clienții, utilizatorii, managerul, Product Owner și echipa de dezvoltare participă la prima parte a întâlnirii ca să identifice obiectivele și funcționalitățile pentru următorul sprint. La a doua parte a întâlnirii Scrum Master-ul și echipa de dezvoltare participă pentru a discuta modul în care va fi implementată unitatea de produs pe parcursul acestui sprint.

Ședința zilnică - se desfășoară zilnic pentru a urmări în continuu progresul echipei. De asemenea, servește ca ședință de planificare se discută: acțiunile întreprinse de la ultima ședință până în prezent și acțiunile care trebuie întreprinse până la următoarea ședință, problemele și impedimentele care i-au împiedicat pe membrii echipei să își atingă obiectivele stabilite. Scrum Master-ul/Team Leader-ul conduce ședința, care durează aproximativ 15 minute. Fiecare ședința ia loc în conformitate cu următoarele principii:

- Ședința se începe la timp și prezența completă este necesară;
- Oricine poate fi prezent în timpul ședinței, dar participă doar acei care participă direct în procesul de dezvoltare;
- Întâlnirea durează 15 minute, indiferent de numărul de participanți;
- Fiecare participant vorbește pe rând, nimeni nu-i întrerupe;
- Aceste ședințe ar trebui să aibă loc zilnic în același loc la aceeași oră.

Ședința de revizuire a sprint-ului și de retrospectivă - o ședință care marchează ultima zi a sprint-ului și reprezintă o oportunitate pentru echipa de a reflecta la ceea ce a fost făcut. Ca parte a acestei ședințe, Scrum Master-ul prezintă rezultatele sprintului și încurajează echipa să îmbunătățească procesul de lucru. Participanții la ședință evaluează rezultatul și iau decizii cu privire la cursul acțiunilor preconizate. Ședința de revizuire a sprint-ului și de retrospectivă trebuie să includă următoarele:

- Practicile care au decurs bine în timpul sprintului;
- Analiza dificultăților, problemelor și nemulțumirilor întâmpinate;
- Aspectele învățate de echipa în timpul sprintului;
- Ce ar putea fi îmbunătățit în timpul sprintului următor?

2. Introducere

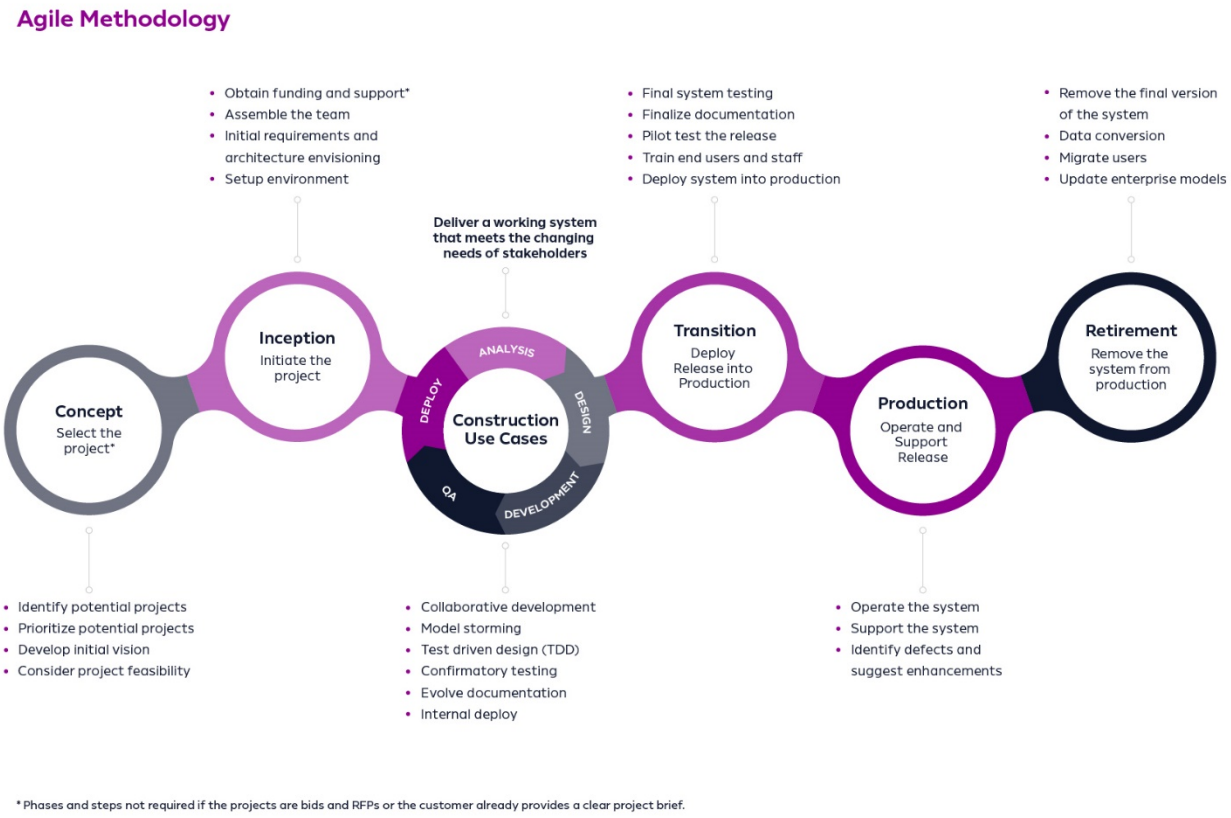
Furnizorul are o experiență vastă în implementarea proiectelor similare. Această experiență se reflectă în fluxurile și metodologiile de implementare, dar și în posibilitatea de a asigura o echipă dedicată care este capabilă să îndeplinească toate obiectivele proiectului.

În continuare sunt prezentate metodologia și procedurile care vor fi implementate în cadrul proiectului, structura acestora, precum și rezultatele preconizate.

Metodologia Agile de dezvoltare/implementare este o metodologie flexibilă de implementare a proiectelor din sectorul IT concepută pentru a răspunde nevoilor de afaceri ale Beneficiarului printr-o abordare iterativă. Cadrul în cauză susține mai multe tipuri de dezvoltare potrivite pentru următoarele scenarii: când Beneficiarul dorește o nouă soluție IT și cazul când are nevoie ca Prestatorul s-o îmbunătățească pe cea existentă.

Serviciile noastre sunt centrate pe individ, beneficiar și utilizatorul final. Prioritatea noastră este să satisfacem nevoile beneficiarului și să livrăm soluțiile la timp. Livrările de software funcționale (modul) sunt efectuate periodic la intervale scurte în conformitate cu prioritățile din backlog.

Metodologia noastră agilă este adaptată la proiecte IT cu preț fix sau contractele în funcție de timp și materiale.



3. Scrum - Prezentare de Ansamblu

Principiul care stă la baza metodologiei noastre de lucru (**Scrum**) constă în dezvoltarea incrementală a aplicațiilor software, păstrând, de asemenea, o listă transparentă a cerințelor și a tuturor schimbărilor care trebuie sa fie făcute produsului (Backlog). Prin utilizarea metodologiei de dezvoltare a software-ului agil, riscurile de dezvoltare sunt reduse, împreună cu intervalul de timp de execuție. Prin urmare, proiectele sunt implementate într-o formă extrem de flexibilă și calitatea livrărilor crește cu fiecare sprint.

Scrum poate fi descris ca un „framework în cadrul căruia poți folosi diverse procese și tehnici”, mai degrabă decât un singur proces sau o tehnică pentru dezvoltarea produsului. Framework-ul Scrum se bazează în mare parte pe echipă, inclusiv roluri asociate, evenimente, artefacte și reguli.

Fiecare proiect este livrat într-o manieră extrem de flexibilă și iterativă în care la sfârșitul fiecărui sprint de lucru există o livrare tangibilă pentru afaceri. Acest lucru poate fi văzut în următoarea diagramă:

Cerințele care stau la baza proiectului sunt incluse în ceea ce se numește un backlog

de proiect (**Project Backlog**) și este actualizat regulat. Caracteristicile care sunt asociate cu aceste cerințe sunt denumite **User Stories/Use Cases** (conform Termenilor de Referință pentru fiecare misiune specifică).

Lucrul este împărțit pe serii, fiecare dintre ele având o durată de la o săptămână la 4, în timpul cărora echipa ordonează **Use Cases** în dependența de prioritatea sarcinilor în realizarea proiectului la fiecare ciclu sau iterație (**Iteration**). Acest subset de Use Cases din Project Backlog constituie baza **Iteration Backlog** planificată pentru livrare timp de 1-4 săptămâni. În cadrul Scrum, există 3 ședințe organizate în timpul unei iterații, plus o ședință zilnică de stand-up în care participă echipa de dezvoltare, Scrum Master și Product Owner. La începutul unui sprint, caracteristicile care trebuie dezvoltate în timpul sprintului sunt decise în cadrul ședinței de planificare a sprintului. La sfârșitul Iterației are loc o altă întâlnire, **revizuirea iterației și ședința de retrospectivă**, în cadrul căreia echipa examinează produsul și demonstrează utilizarea software-ului, precum și discută despre îmbunătățirea procesului de iterație. După finalizarea sprint-ului, următorul set de Use Cases este selectat din Project Backlog și procesul începe din nou.

4. Formarea echipei

Echipa de proiect constă dintr-un grup de experți-cheie, capabili să livreze independent cerințele Beneficiarului. Prin urmare, echipa este inter-funcțională datorită abilităților și cunoștințelor în domeniile de date, instrumentelor și infrastructurii. Pentru proiectele guvernamentale, de obicei, echipa este deja definită și experții-cheie, care vor fi implicați în proiect, se cunosc la etapa de pre-vânzare. În Scrum sunt definite 3 roluri principale:

Product Owner - reprezintă vocea și interesele Beneficiarului. Acesta este responsabil pentru proiectarea, managementul, controlul și prezentarea backlog-ului de proiect. Product Owner-ul stabilește prioritățile și are dreptul de decizie finală asupra sarcinilor nerezolvate.

Scrum Master – este reprezentantul Furnizorului, care acționează ca manager de proiect, și are grijă ca procesul de dezvoltare să deruleze în conformitate cu metodele, valorile și regulile Scrum. El interacționează atât cu echipa de dezvoltare, cât și cu Product Owner și cu părțile interesate. El va fi, de asemenea, responsabil de eliminarea oricăror impedimente ca productivitatea echipei să fie permanent la un nivel înalt.

Echipa de dezvoltare – grupul de specialiști care se ocupă de dezvoltarea produselor. Echipa are autoritatea de a decide ce măsuri trebuie luate pentru a rezolva sarcina atribuită fiecărui sprint și are dreptul de a se autoorganiza în aceste scopuri. Echipa de dezvoltare poate consta din:

- Dezvoltator baze de date;
- Specialist în integrare / Dezvoltator software;
- Specialist DevOps / Dezvoltator software;
- Software Tester;
- Trainer.

În abordarea agilă, un aspect deosebit de important este reprezentat de implicarea utilizatorilor, clienților și părților interesate în procesul de dezvoltare. Părțile interesate (**Stakeholders**) oferă feedback cu privire la rezultatele fiecărui sprint pentru a ajusta și îmbunătăți procesele de lucru, printre ele se numără:

- Utilizatorii – grupul de persoane care vor folosi produsul final în cadrul organizației sau din extern.
- Clienții - cei care definesc obiectivul proiectului și sunt implicați în procesul de dezvoltare numai la etapa de evaluare a unui sprint;

- Sponsorii – oamenii sau organizațiile care finanțează un proiect. De asemenea, participă la stabilirea obiectivului și condițiilor de muncă. Ei reprezintă partea interesată cui toți cei implicați în proiect îi raportează.

5. Iterația Zero

În **Iterația Zero**, echipa explorează ideile de produs, nevoile Beneficiarului, practicile de dezvoltare, arhitectura hardware și software. Echipa își creează viziunea asupra funcționalităților necesare și a dezvoltărilor necesare. Aceștia convin asupra obiectivelor de dezvoltare, inclusiv nevoilor pieței, nevoilor afacerii, conținutului produsului și efortului necesar pentru dezvoltarea produsului. Echipa cu ajutorul Product Owner-ului elaborează planul inițial. Scopul **Iterației Zero** este dobândirea unei înțelegeri clare a următoarelor aspecte:

- Backlog-ul produsului și prototipurile;
- Arhitectura tehnică;
- Legătura dintre timp și bani.

5.1 Instrumente de lucru și training-uri

Echipa va folosi iterația zero pentru a seta instrumentele și informație necesare, astfel încât iterația să se poată concentra pe dezvoltarea prime părți a produsului. Unele dintre activitățile care vor fi furnizate de Beneficiar sunt următoarele:

- Configurarea instrumentelor, infrastructurii pentru mediile de testare și producție;
- Crearea repozitoriului de cod sursă, sistemului de urmărire a problemelor, mediului CI / CD, sistemului de gestionare a sarcinilor;
- Configurarea instrumentelor pentru UAT (User Acceptance Testing) și Automation Testing;
- Configurarea convențiilor și instrumentelor de raportare.

6. Faza inițială. Warm-up Iteration

Această iterație presupune definirea sistemului care trebuie construit. Product Owner-ul va crea **Backlog-ul proiectului** sau **caietul de sarcini cu cerințele de software (SRS)**, care va conține cerințele cunoscute la etapa actuală sau specificate în Termenii de referință pentru proiect. Prioritățile sunt atribuite cerințelor și este evaluat efortul necesar pentru implementarea acestora. De asemenea, echipa de proiect este formată în această fază împreună cu instrumentele și resursele necesare.

6.1 Colectarea inițială a cerințelor

Etapa inițială de evaluare a cerințelor presupune stabilirea cerințelor funcționale ale sistemului, pornind de la analiza nevoilor, generată de activitatea Beneficiarului și coroborarea

acestor informații cu diferite tipuri de date legate de: infrastructura tehnică și de securitate, utilizatorii sistemului, cazurile de utilizare, fluxurile de lucru, procesele legale și reglementările interne ale Beneficiarului, procedurile specifice de lucru și standardele internaționale aplicabile în domeniu. De asemenea, evaluarea cerințelor va ține cont de Termenii de referință pentru fiecare proiect aparte.

Activitățile de evaluare a cerințelor se desfășoară la sediul Beneficiarului (pentru a identifica necesitățile specifice, procesele de lucru și reglementările interne) și la sediul Furnizorului, pentru a integra toate datele, a redacta documentul de evaluare și a analiza componentele tehnice și detaliile de implementare.

Rezultatele etapei de evaluare a cerințelor constituie un set de specificații funcționale, convenite în comun cu Beneficiarul produsului și alte părți interesate.

Evaluarea cuprinde:

- Evaluarea situației actuale din instituția Beneficiarului;
- Stabilirea obiectivelor generale ale proiectului;
- Stabilirea obiectivelor specifice prin rafinarea obiectivelor generale;
- Determinarea părților implicate (entități organizaționale care fac parte din organizația Beneficiarului, implicate în utilizarea și administrarea produsului care urmează a fi livrat);
- Configurarea cerințelor funcționale pentru fiecare parte implicată;
- Constituirea arhitecturii generale - stabilirea modulelor / componentelor necesare.

Faza include și redactarea raportului de evaluare care va include Backlog-ul proiectului prezentat de Product Owner.

Backlog-ul proiectului va conține toate cazurile de utilizare (Use Cases) ale produsului. Product Owner-ul este responsabil de definirea cazurilor de utilizare și atribuirea fiecăruia priorități de livrare. Prioritățile pot fi influențate și de diversitatea cazurilor de utilizare sau de valoarea comercială.

Cazurile de utilizare vor descrie în detaliu setul de caracteristici care vor satisface fiecare cerința funcțională. Cazurile de utilizare cu prioritate ridicată, care sunt planificate pentru următorul Sprint, trebuie să aibă suficient de multe detalii pentru a putea fi implementate de echipa timp de 4 săptămâni.

Product Owner-ul va asigura ca fiecare caz de utilizare să corespundă următoarelor criterii:

- Obiectivul cazului de utilizare;
- Actorul principal - cine va avea acces la acest caz de utilizare;
- Nivelul - la ce tip de informații și funcții are acces actorul principal;

- Scurtă descriere/Obiectiv descrie funcția pe care actorul principal vrea să o îndeplinească;
- Condițiile preliminare (Preconditions) specifică că sistemul se va asigura că este adevărat înainte de începerea cazului de utilizare. De regulă, o precondiție indică faptul că un alt caz de utilizare a fost rulat pentru a-l configura
- Declanșările (Triggers) - evenimentele care determină inițierea cazului de utilizare.
- Fluxul principal (Main Flow) - cazul de utilizare în care nu se întâmplă nimic incorect (norma dorită). Scenariul constă dintr-o succesiune de pași. Acestea sunt interacțiunile necesare între actori și soluția pentru atingerea obiectivului dorit.
- Fluxul alternativ (Alternative Flow) - variații de la fluxul principal
- Postcondițiile - condițiile care trebuie să fie îndeplinite înainte de folosirea cazului de utilizare;
- Cerințe nefuncționale includ utilitate, fiabilitate, securitate, flexibilitate.
- Criterii de acceptare - set de declarații, fiecare cu un rezultat clar de admitere/eșec, care specifică atât cerințele funcționale (de exemplu, funcționalitate minimă comercializabilă), cât și nefuncționale (de exemplu, calitate minimă), aplicabile la etapa actuală a integrării proiectului.

Este important de menționat că pentru a crește exactitatea oricărei estimări, efortul implicat va crește exponențial. Pentru această misiune specifică, suntem interesați doar de descrierea și estimarea corectă. Având control asupra Sprint-urilor, putem mai bine analiza și mai repede reacționa la abaterile în estimare.

6.2 Arhitectura inițială de nivel înalt

Definirea Arhitecturii/Proiectarea la nivel înalt - aici se realizează proiectarea la nivel înalt a sistemului; dacă sistemul există deja, sunt discutate modificările necesare pentru a implementa cerințele, precum și problemele pe care le implică aceste modificări.

Echipa de dezvoltare va discuta și apoi va schița o arhitectură potențială pentru sistem. Această arhitectură va evolua în timp, nu va fi încă foarte detaliată (trebuie doar să fie suficient de bună deocamdată). Scopul este de a identifica o strategie arhitecturală, nu de a scrie documente complexe și sofisticate.

6.3 Livrabile

#	Artefacte	Responsabil
1	Schița documentației tehnice – inclusiv documentația de arhitectură a sistemului (descrierea modelelor în limbajul UML, unde este descrisă destul de detaliat arhitectura	Scrum Master Echipa de dezvoltare

	sistemului). Acest document va fi dezvoltat iterativ.	
2	Backlog-ul de soluții	Product Owner/Scrum Master/Team Lead
3	Planul de release	Product Owner/Scrum Master/Team Lead
4	Schița planurilor de User Acceptance Testing, Testării Load & Stress și Automation Testing	QA/Software Tester
5	Proiectul planului de mentenanță	Scrum Master/Team Lead

Planul de testare funcțională și testarea planului de acceptanță constituie anexe la raportul final de evaluare și constituie documente pe baza cărora trebuie verificate, validate și acceptate rezultatele proiectului.

7. Iterațiile de dezvoltare(Sprints)

Sarcinile de dezvoltare sunt organizate în sprinturi, iar conținutul sprinturilor se definește pe baza Ședințelor de planificare (Sprint Planning Meetings). Pe parcursul acestei întâlniri, Product Owner-ul informează echipa despre sarcinile nerezolvate pe care dorește să le abordeze. În consecință, echipa stabilește câte dintre aceste sarcini pot fi îndeplinite până la următorul Sprint. Sarcinile selectate nu pot fi modificate în timpul sprintului. La final, echipa prezintă funcțiile dezvoltate și modul în care este utilizat produsul intermediar obținut.

Datorită acestei metode de organizare a echipelor de dezvoltare, echipele sunt independente și autoorganizate, cu o comunicare verbală/scrisă transparentă între toți membrii echipei și diferite departamente ale proiectului.

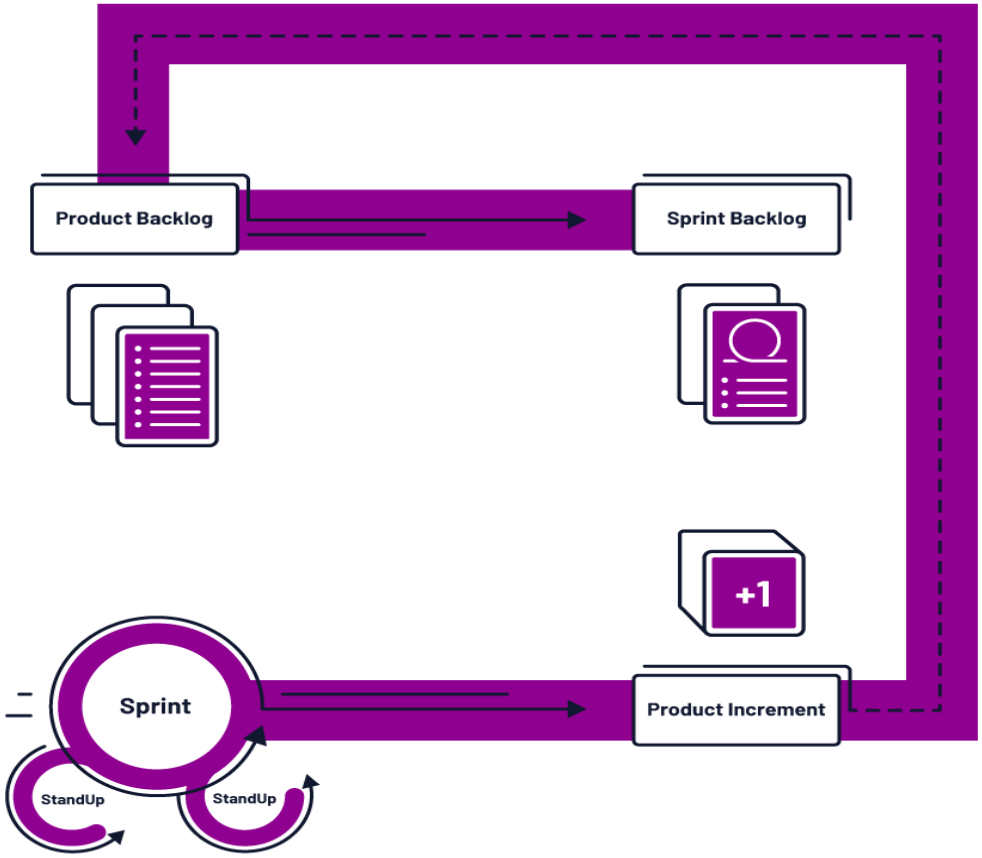
În general, se acceptă faptul că, pe parcursul dezvoltării unui proiect, Product Owner-ul își poate schimba părerea cu privire la așteptările față de produsul software. Astfel de modificări sunt imprevizibile și nu sunt ușor de ajustat în cadrul proiectelor unde se aplică metodele tradiționale de dezvoltare a software-ului.

7.1 Ședințele de planificare ale sprinturilor (Sprint Planning Meetings)

Ședința de planificare a sprintului se desfășoară înainte de începutul fiecărui Sprint și permite Product Owner-ului și Echipei de Dezvoltare să discute cerințele și lucrările necesare pentru următorul release. Această etapă a procesului Scrum se concentrează pe determinarea scopului de livrare al unui Sprint și definește Backlog-ul Sprintului.

Ședința de planificare a Sprintului nu trebuie să dureze mai de mult 6 ore (pentru un Sprint care durează 4 săptămâni). Product Owner-ul și Scrum Master-ul/Team Leader-ul de echipă sunt responsabili de actualizarea Backlog-ului Proiectului înainte de ședința de

planificare. Aceasta include clarificarea, stabilirea priorităților și, în unele cazuri, investigarea fezabilității și interdependenței cazurilor de utilizare selectate. De asemenea, această activitate trebuie să ia în considerare orice datorie tehnică moștenită de la Sprinturile anterioare.



7.1.1 Specificațiile privind cerințele comerciale

Prima parte a ședinței de planificare a sprint-ului are ca scop transformarea funcționalităților selectate din Backlog-ul proiectului într-un obiectiv realizabil pentru acest sprint. Product Owner-ul face parte din acest proces și stabilește prioritatea sarcinilor. Astfel Product Owner-ul are posibilitatea de a comunica planul de livrare, de a oferi contextul și prioritățile itemilor și de a răspunde la întrebările echipei de proiect care ar ajuta la descompunerea funcțională și estimarea.

Această parte a ședinței nu trebuie să dureze mai mult de ¼ din timpul alocat ședinței. Echipa va primi o copie a backlog-ului de proiect înainte de ședința de planificare a Sprintului pentru a avea timp să ia în considerare soluțiile posibile discutate în timpul ședinței și pentru a pregăti întrebările de clarificare pentru Product Owner. La această sesiune participă:

- Product Owner;
- Scrum Master;
- Echipa de dezvoltare;
- Testerii de software.

7.1.2 Specificațiile tehnice și evaluarea sarcinilor

A doua parte a ședinței de Sprint Planning este tehnică și, de regulă, ia loc fără participarea Product Owner-ului. Aceasta presupune decompoziția soluției și estimarea efortului pentru dezvoltarea și testarea funcționalităților pentru release-ul nou.

Fluxul general al activității în această etapă este descris după cum urmează:

- Selectarea cazurilor de utilizare (stabilită de Product Owner);
- Crearea diagramelor UML sau diagramelor de secvențe detaliate pentru fiecare caz de utilizare;
- Crearea modelului logic de date;
- Determinarea task-urilor și subtask-urilor pentru livrarea cazurilor de utilizare;
- Aprobarea ordinii și Dependențelor Tehnice;
- Estimarea efortului necesar pentru finalizarea fiecărei sarcini - o activitate estimată nu va depăși termenul de 8 ore;
- Evaluarea riscurilor legate de estimare a sarcinilor;
- Repartizarea activităților între membrii echipei de dezvoltare.

Activitățile/task-urile mari trebuie împărțite în subtask-uri, de preferință nu mai mari de 8 ore, iar sarcinile care implică așteptare trebuie împărțite în subsarcini separate. Activitățile de cercetare ar trebui să aibă o estimare înaltă a riscului.

Pentru cazuri complexe de utilizare, cu un număr mare de interdependențe, poate fi necesar de efectuat descompunerea și estimarea activităților pe parcursul a 2 zile permițând membrilor echipei să se consulte cu părțile externe privitor la fezabilitatea activităților și obținerea de ajutor în procesul de estimare.

7.1.3 Structura cazurilor de utilizare și a task-urilor

Activitățile adăugate în tracker-ul de sarcini vor avea următoarea structură predefinită:

- Titlu cu o scurtă descriere - Ex: [DevBE];[DevFE];[DevOps];[QA/Tester];
- Descrierea sarcinii cu următoarele componente: Diagrama de flux/diagrama de secvențe/model de date logice;
- Artefactele necesare pentru executarea sarcinilor;
- Criteriile de acceptare;
- Tipul sarcinii: funcționalitate, eroare, epic etc.;
- Timpul estimat;

- Termenul limită (prioritate în planificare): data;
- Prioritate: scăzut, mediu, înalt, critic;
- Sprint corespunzător;
- Responsabil: membru al echipei;
- Task-ul/subtask-ul afiliat/interdependent;
- Status: neprocesat, nou, în curs de dezvoltare, dezvoltat, testat, lansat, finalizat.

7.2 Ședințe zilnice Scrum (Daily Stand-Ups)

Întâlnirea zilnică pune un accent constant pe informarea Product Owner-ului despre progresele și livrările în cadrul iterației. Ședința trebuie să fie informativă, interactivă și să alinieze obiectivele echipei legate de ceea ce este in proces de implementare, cine se ocupă de asta și statusul curent. Product Owner-ul, Scrum Master-ul/Team Leader-ul și echipa de proiect participă la această întâlnire care durează 15 minute. Toți participanți răspund la următoarele trei întrebări:

- Ce am realizat ieri?
- Ce voi realiza astăzi?
- Ce impedimente pot împiedica să ating obiectivele de astăzi?

Obiectivul Scrum Master-ului/Team Lead-ului pe durata acestor ședințe este de a elimina orice impediment identificat de echipă.

7.3 Ședințele de Retrospectivă și Revizuire a Sprint-ului

La sfârșitul Sprintului are loc o ședință de revizuire a sprint-ului pentru a evalua incrementarea produsului și pentru a actualiza Backlog-ul Proiectului, după necesitate. Echipa de proiect și Product Owner-ul evaluează ce a fost făcut în cadrul Sprint-ului. Pe baza acestei activități și a oricăror modificări ale Backlog-ului Produsului în timpul Sprint-ului, participanții evaluează următorii pași care ar putea optimiza valoarea. Aceasta este o întâlnire informală, prezentarea incrementării de produs fiind destinată pentru obținerea feedback-ului și să favorizeze colaborarea. Pentru Sprint-uri cu durata de o lună aceasta întâlnire durează până la 4 ore.

Scrum Master-ul/Team Leader-ul se va asigura ca ședința să aibă loc și ca participanții să înțeleagă scopul acesteia. Scrum Master-ul îi ghidează pe toți pe durata ședinței.

Retrospectiva Sprint-ului include următoarele elemente:

- Participă echipa de proiect și Product Owner;
- Product Owner-ul explică ce sarcini au fost „Realizate” și ce nu a fost „Realizat”;
- Echipa de dezvoltare discută despre ce a decurs bine în timpul Sprint-ului, cu ce probleme s-a confruntat și cum au fost soluționate;

- Echipa de Dezvoltare demonstrează activitățile „Realizate” și răspunde la întrebări despre incrementarea de produs;
- Product Owner-ul discută Backlog-ul actualizat. Acesta estimează termenii probabili de finalizare a proiectului bazați pe progresul actualizat (dacă este necesar);
- Întregul grup evaluează ce trebuie să facă în continuare, astfel încât Revizuirea Sprint-ului să ofere o contribuție valoroasă pentru planificarea Sprint-ului următor.

Rezultatul retrospectivei de Sprint este un Backlog al Proiectului actualizat pentru următorul Sprint. Backlog-ul proiectului poate fi, de asemenea, ajustat pentru a face față noilor oportunități.

Retrospectiva Sprint-ului este o oportunitate pentru Echipa de Proiect de a se autoanaliza și de a crea un plan pentru îmbunătățiri care vor fi adoptate în următorul Sprint.

Scrum Master-ul/Team Leader-ul participă ca un membru egal al echipei la ședință. Scopul Retrospectivei de Sprint este:

- Verificarea progresului ultimului Sprint cu privire la oameni, relații, procese și instrumente;
- Identificarea și ordonarea elementelor majore care au mers bine și îmbunătățirilor posibile;
- Crearea unui plan pentru implementarea îmbunătățirilor la modul în care echipa de proiect își îndeplinește activitatea.

7.4 Revizuirea cerințelor detaliate

Reprezintă revizuirea detaliată a cerințelor și include:

- Stabilirea și definirea grupurilor de lucru;
- Identificarea și analiza infrastructurii tehnice și de comunicare ale Beneficiarului;
- Detalierea fluxurilor în procese / activități adecvate;
- Trasabilitatea activităților identificate cu cerințele părților implicate, identificate în faza de colectare a cerințelor inițială, pentru a asigura acoperirea întregului domeniu de activitate;
- Identificarea cerințelor funcționale bazate pe activități;
- Trasabilitatea între cerințele funcționale și funcționalitățile aplicației standard;
- Identificarea nevoilor de schimbare a funcționalităților standard și de dezvoltare a celor suplimentare;
- Identificarea potențialelor probleme și riscuri care pot afecta implementarea / utilizarea sistemului (inclusiv probleme potențiale de incompatibilitate între diferite module);
- Identificarea măsurilor de atenuare a riscurilor și de remediere a problemelor potențiale care pot fi prevăzute în această fază;

- Identificarea cerințelor de instruire pentru utilizatorii finali.

7.5 Designul detaliat al soluției (Model Storming)

Arhitectura funcțională detaliată este definită în timpul iterațiilor de dezvoltare. Această activitate implică redactarea scenariilor cazurilor de utilizare detaliate, la nivelul cazurilor de utilizare particulare, inclusiv rolurile, constrângerile și regulile asociate fiecărui caz, precum și potențiale excepții.

În același timp, este definită arhitectura tehnică, inclusiv o descriere a infrastructurii de asistență (servere, stații de lucru, infrastructuri și protocoale de comunicare, surse de date, stocare de date etc.).

De asemenea, dezvoltarea modelului informațional presupune definirea arhitecturii de date a sistemului, la nivel logic și fizic.

Modulele de sistem sunt proiectate și descrise în această fază.

Proiectarea sistemului poate identifica mai multe soluții, urmărind simplitatea și eficiența realizării și implementării cerințelor Beneficiarului, respectând totodată restricțiile tehnice, organizaționale, financiare sau legale.

Procesul de proiectare pornește de la nevoile și prioritățile Beneficiarului, având în vedere natura esențială a implicării utilizatorilor sistemului, cu scopul înțelegerii corecte a proceselor de lucru și a acceptării de către utilizatori a noului sistem.

7.6 Testarea confirmativă QA

Echipa realizează o evaluare obiectivă pentru a asigura calitatea produsului dezvoltat. Aceasta include depistarea defectelor, validarea funcționalității sistemului precum este proiectat și verificarea respectării cerințelor. Toți actorii implicați sunt la fel de responsabili pentru calitatea produsului și succesul proiectului.

Acest lucru înseamnă că testarea confirmativă este realizată de întreaga echipă, inclusiv de membrii echipei a căror expertiză principală poate fi în programare, analiză de afaceri, administrare de baze de date sau sisteme, nu doar de testeri desemnați sau de profesioniștii de asigurare a calității.

Membrii echipei de testare software nu se limitează la realizarea acestei activități - vor colabora cu Product Owner în privința cerințelor produsului și vor lucra cu alți membri ai echipei pentru a dezvolta un cod de înaltă calitate care să îndeplinească aceste cerințe.

În timpul procesului de dezvoltare în cadrul Sprint-ului, soluțiile evoluează prin colaborarea dintre echipele auto-organizate, inter-funcționale, care descoperă - adesea prin încercare și eroare - cele mai bune procese, practici și instrumente de utilizat în diferite contexte.

Dezvoltarea agilă are o abordare bazată pe testare primară, mai degrabă decât o abordare de testare la sfârșit a dezvoltării tradiționale. Testarea și codarea agilă se realizează în mod incremental și interactiv, construind fiecare funcționalitate până când oferă o valoare suficientă pentru a fi lansată în producție.

Deoarece testarea agilă se bazează pe feedback-ul obișnuit al utilizatorului final sau al Product Owner-ului, din acest aspect se diminuează posibilitatea de a întâmpina o problemă comună pe care o au multe echipe de software, care creează o soluție greșită, deoarece echipa interpretează greșit o caracteristică și aliniază ceea ce văd cu expertiza lor de dezvoltare, mai degrabă decât ceea ce spune cerința sau ce dorește utilizatorul final.

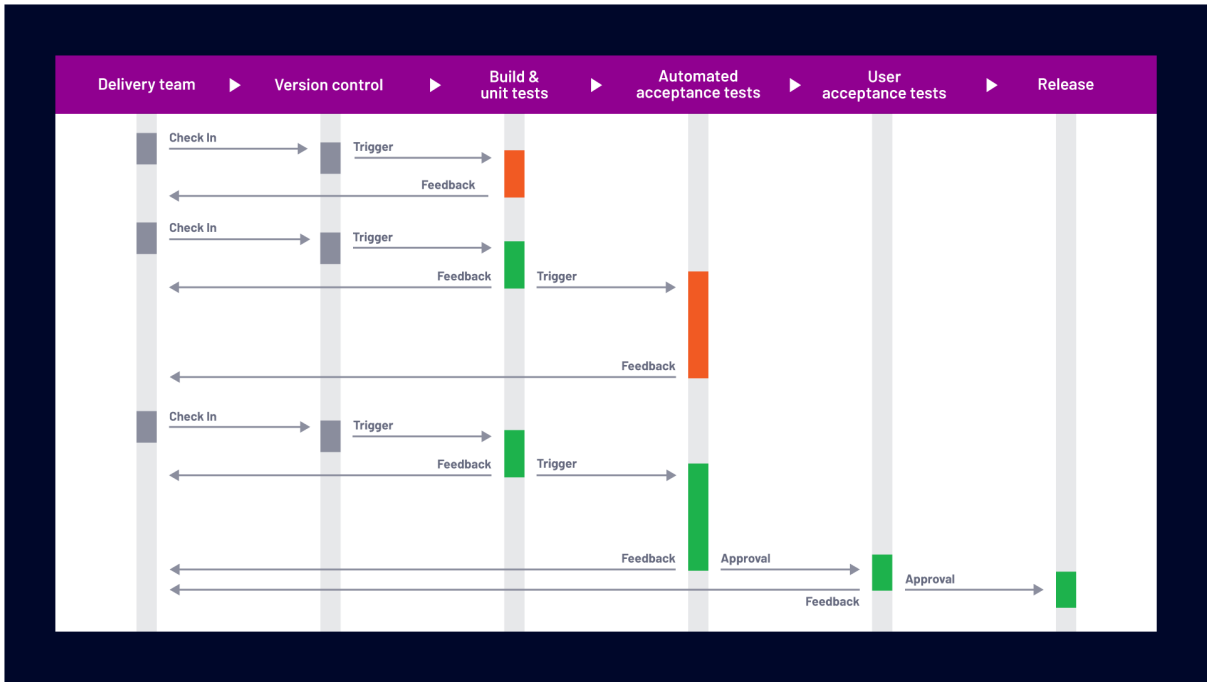
7.6.1 Testarea ciclului de viață cu aplicarea QA

Spre deosebire de metodologia Waterfall, Testarea Agile nu este secvențială - sau efectuată după o fază de codificare - ci reprezintă un proces continuu. Testarea continuă este una dintre mai multele activități continue care se desfășoară simultan pe majoritatea Sprint-urilor, inclusiv:

- Compilare continuă;
- Integrare continuă (CI);
- Livrare continuă (CD);
- Implementare continuă.

În mod ideal, pentru această dezvoltare specifică, compilările și testările se vor produce zilnic. Pentru a implementa acest lucru, echipele vor folosi un proces de integrare continuă și implementare continuă (CI / CD). CI / CD limitează posibilitatea construirii eșuate în ziua unei note de lansare.

Integrarea continuă este o practică în care membrii unei echipe de dezvoltare utilizează un sistem de control al versiunilor și își integrează modificările de cod frecvent în aceeași locație, cum ar fi o ramură principală. Fiecare modificare este construită și verificată prin teste și alte verificări pentru a detecta cât mai repede eventualele erori de integrare. Odată cu automatizarea compilării, software-ul se generează automat, folosind instrumente precum Makefiles sau Ant, mai degrabă decât atunci când un dezvoltator invocă manual compilatorul.

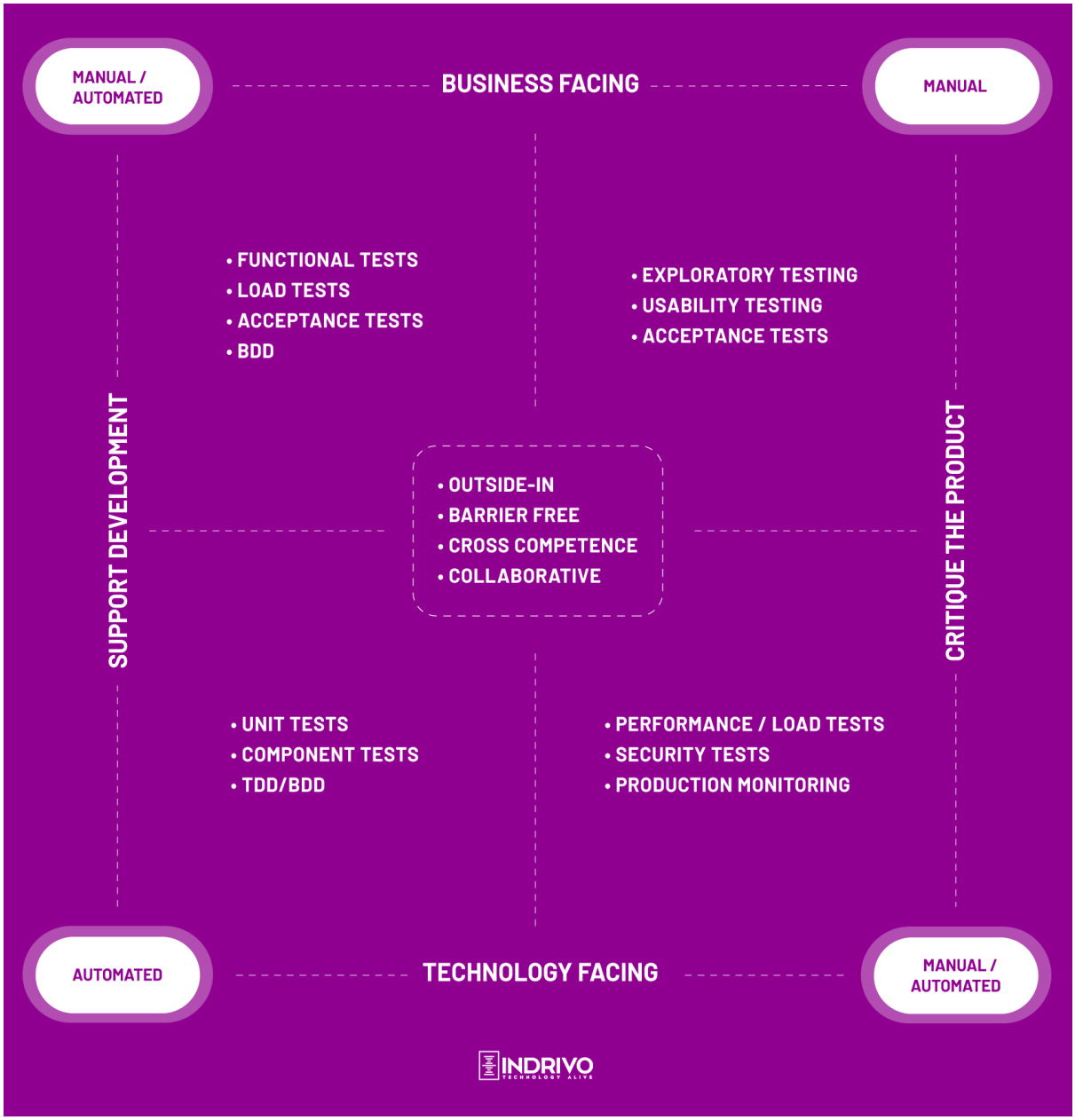


În ultima etapă a unei etape CI / CD, odată ce o aplicație trece toate testele necesare, este apoi lansată în producție. Aceasta reprezintă livrarea produsului către Product Owner.

7.6.2 Cadranele de testare

Deoarece Agile este o metodologie de dezvoltare iterativă, testarea și scrierea de cod sunt realizate în mod incremental și interactiv, unde funcțiile pot evolua ca răspuns la modificarea cerințelor Beneficiarului.

Testările agile acoperă toate tipurile de teste, inclusiv testele de unitate, de integrare, funcționale, de încărcare și de performanță. Următoarea diagramă este un model util pentru echipele multifuncționale de dezvoltare agilă pentru planificarea și executarea activităților de testare:



Cele patru cadrane sunt descrise mai detaliat mai jos:

Cadranul 1: Acestea sunt teste orientate către tehnologie care ghidează dezvoltarea, cum ar fi testele unitare, testele API, testarea serviciilor web și testele de componente care îmbunătățesc designul produsului. Testele din primul cadran sunt adesea asociate testării automate și integrării continue.

Cadranul 2: Acestea sunt teste orientate spre funcționalități de business care ghidează dezvoltarea, cum ar fi cele utilizate pentru testarea funcțională, prototipuri și simulări care asigură că produsele software sunt aliniate corespunzător afacerii. Testele în Q2 sunt adesea asociate atât cu testări automate, cât și manuale.

Cadranul 3: Acestea sunt teste orientate către afaceri, utilizate pentru evaluarea sau critica produsului. Q3 acoperă teste precum teste exploratorii, teste bazate pe scenarii, teste de utilizabilitate, teste de acceptare a utilizatorilor și teste alfa / beta și pot implica demonstrații

de produse concepute pentru a obține feedback de la utilizatorii reali. Testele în Q3 sunt adesea asociate cu testarea manuală.

Cadranul 4: Acestea sunt teste orientate către tehnologie, utilizate pentru evaluarea sau critica produsului. Q4 acoperă teste precum teste de performanță, încărcare, stres și scalabilitate, teste de securitate, întreținere, gestionarea memoriei, compatibilitate și interoperabilitate, migrare de date, infrastructură și teste de recuperare. Aceste teste sunt adesea automatizate.

7.6.3 Dezvoltarea bazată pe testul de acceptare (ATDD)

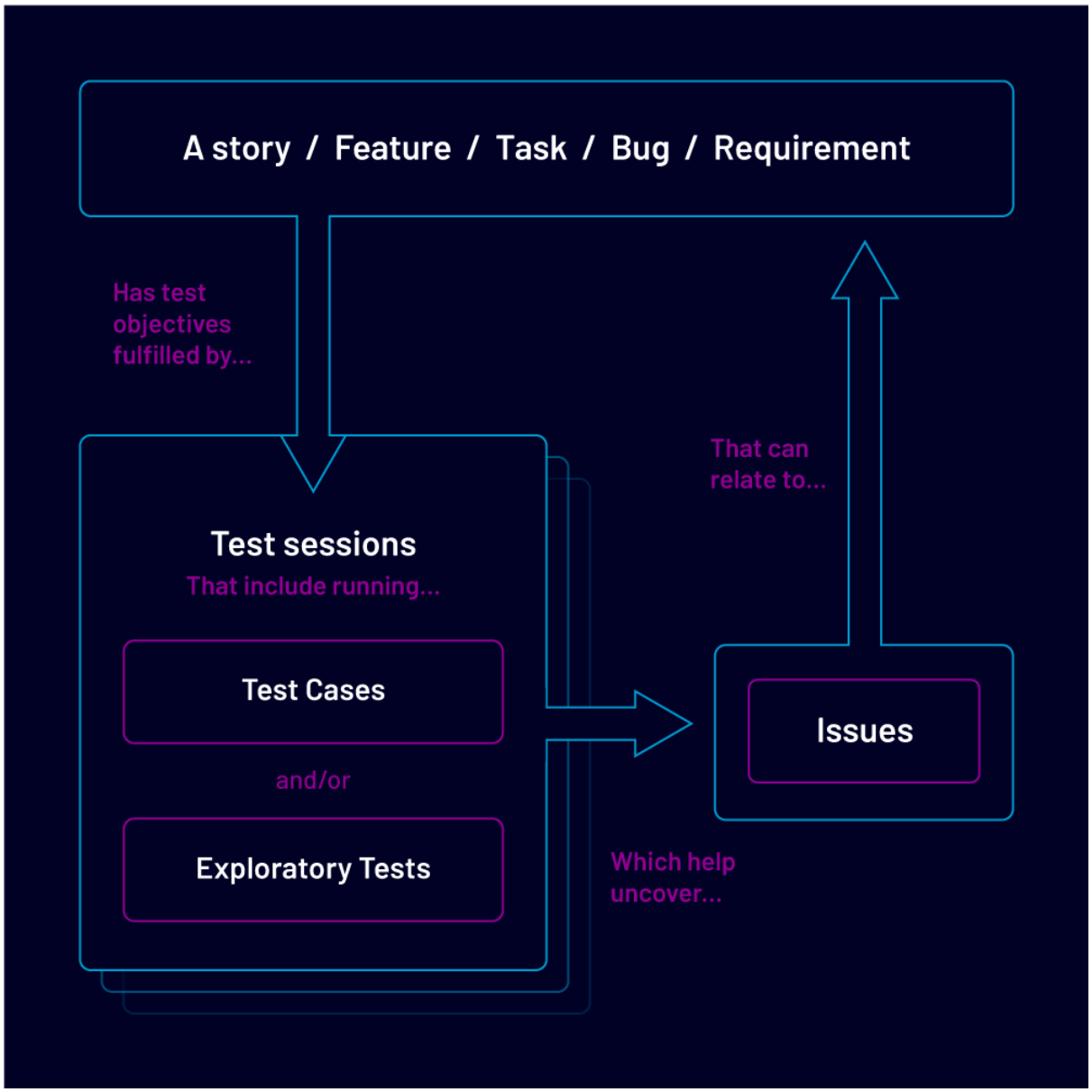
Dezvoltarea bazată pe testul de acceptare (ATDD) este o altă îmbunătățire a dezvoltării bazate pe teste care promovează colaborarea între Product Owner, testeri și dezvoltatori pentru a defini criteriile de acceptare automate înainte de începerea codificării. ATDD și TDD sunt tehnici complementare: ATDD ajută la descrierea obiectivelor de afaceri la nivel înalt, în timp ce TDD ajută dezvoltatorilor să le implementeze ca cerințe. ATDD ajută să se asigure că toți membrii proiectului înțeleg ce este implementat, deoarece eșecul testelor ATDD oferă un feedback rapid și semnalează când cerințele nu sunt îndeplinite.

O parte cheie a testelor ATDD este că acestea sunt rulate automat de fiecare dată când se face o modificare a codului sursă. Pe lângă testarea aplicației, testele de acceptare automată sunt utile pentru măsurarea progresului pe care echipa de proiect îl face, deoarece, pe un proiect agil, software-ul funcțional este considerat a fi singura măsură obiectivă a progresului.

7.6.4 Testarea bazata pe sesiune (Session-Based Testing)

Testarea bazată pe sesiune este un tip de testare exploratorie structurată care impune specialiștilor de testare să identifice obiectivele testării și să-și concentreze eforturile de testare pe îndeplinirea acestora. Sesiunile de testare diferă de cazurile de testare în două moduri:

- mai multe teste pot fi efectuate într-o singură sesiune;
- ca și cazurile de testare, datorită sesiunilor de testare putem determina cine a testat ce, când și de ce au fost efectuate testele.



Scripturile de test pre-scrise pot fi rulate în timpul unei sesiuni. Cu toate acestea, deoarece sesiunile de testare subliniază obiectivele testelor asupra anumitor cazuri de testare, testerii sunt încurajați să creeze și să execute mai multe teste pe baza la ceea ce au descoperit și învățat. Acest tip de testare exploratorie este un mod extrem de eficient de a optimiza acoperirea testelor fără a suporta cheltuieli asociate cu scrierea și întreținerea cazurilor de testare.

7.6.5 Testarea Automatizată

Testarea automatizată funcționează pe proiecte agile, prin executarea unui număr mare de teste în mod repetat, ca să ne asiguram că o aplicație nu se sparge de fiecare dată când sunt introduse noi modificări - la nivel de unitate, API și GUI. Pentru multe echipe de dezvoltare Scrum, aceste teste automate sunt executate ca parte a unui proces de construire a integrării continue (CI), unde dezvoltatorii verifică codul într-un depozit partajat de mai multe

ori pe zi. Fiecare check-in este apoi verificat de o construcție automatizată, permițând echipelor să detecteze erorile și conflictele cât mai curând posibil. Instrumentele CI precum Jenkins, Bamboo și Selenium sunt, de asemenea, utilizate pentru a construi, testa și implementa automat aplicațiile atunci când cerințele se schimbă pentru a accelera procesul de implementare.

Automatizarea testelor permite echipelor de dezvoltare Scrum să execute mai multe teste în mai puțin timp, crescând acoperirea și permițând specialiștilor QA să aibă timp pentru testarea exploratorie. Deoarece scripturile de testare automatizată sunt reutilizabile, ele pot fi folosite pentru a face teste mai ample, testând pași repetitivi cu diferite seturi de date, cum ar fi cele pentru compatibilitatea browser-ului sau cross-device-ului.

Printre riscurile de automatizare se numără cele legate de controlul versiunii și de mentenanța scripturilor de testare și a rezultatelor testelor. Alegerea instrumentului adecvat de testare automatizată este importantă pentru a le evita pe acelea care sunt incompatibile cu alte instrumente de testare software din mediul de testare al proiectului. După ce ai un instrument de automatizare a testelor care funcționează bine cu celelalte instrumente de testare, testele automatizate ar trebui să fie, de asemenea, efectuate în mod regulat pentru a oferi feedback în continuare despre starea de sănătate a întregului sistem, de preferință printr-o abordare de integrare continuă descrisă mai sus, ca opus al testării manuale.

7.6.6 Testarea de integrare

Testarea de integrare evaluează funcționalitatea diferitelor module atunci când sunt integrate pentru a forma o singură unitate. Această testare validează tranzițiile fluide între diverse componente integrate ale software-ului. Scopul testării de integrare este de a găsi defecte și defecțiuni între mai multe interfețe ale software-ului.

Testarea integrării include următoarele etape:

1. Testarea integrării începe cu înțelegerea arhitecturii aplicației;
2. Găsirea diferitelor module ale sistemului;
3. Înțelegerea funcționalității fiecărui modul;
4. Evaluarea tranzacției de date între interfețe;
5. Analiza punctelor de intrare și ieșire din sistem;
6. Pregătirea planului pentru testare;
7. Selectarea abordării de testare;
8. Clasificarea modulelor în funcție de nevoile de testare;
9. Identificarea condițiilor de testare pentru cazurile de testare;
10. Proiectarea scenariilor de testare, scripturi și cazuri de testare;
11. Implementarea modulelor alese și efectuarea testării integrării;
12. Executarea cazurilor de testare;

13. Urmărirea defectelor și înregistrarea rezultatelor.

7.7 Planuri de testare

Planul de testare va fi redactat și actualizat pentru fiecare iterație de lansare. Un plan de testare va conține următoarele:

- Scopul testării;
- Consolidarea noilor funcționalități care trebuie testate;
- Tipuri de testare / Niveluri de testare;
- Testarea stresului și încărcării performanței;
- Examinarea infrastructurii;
- Planul riscurilor;
- Planificarea misiunii;
- Note de lansare.

7.7.1 Livrabile

#	Livrabile	Responsabil
1	Strategia și planul de testare iterativă (pentru toate tipurile de testare) în conformitate cu Planul de lansare	Scrum Master/Team Lead Software Tester Product Owner
2	Raportul testării de acceptanță a utilizatorului	Product Owner Scrum Master/Team Lead
3	Raportul testării de stres și performanță	Product Owner Scrum Master/Team Lead

7.8 Actualizarea și evoluția documentației

Pentru a avea o soluție potențial utilizabilă la fiecare iterație, trebuie să păstrăm **documentația de livrare** în sincronizare cu software-ul / soluția - cu alte cuvinte, trebuie să actualizăm documentați pe tot parcursul proiectului. Documentația include de obicei manuale de utilizare, materiale de instruire, manuale de operare, manuale de asistență și prezentări generale ale sistemului. Nu include specificații de cerințe sau specificații de proiectare, cu excepția situațiilor în care o astfel de documentație este necesară sau în negocierile contractuale în care este prevăzută în contract. Documentația livrabilă pentru iterația N este scrisă în timpul iterației N + 1.

7.8.1 Livrabile

#	Livrabile	Responsabili

1	Documentația tehnică - inclusiv documentația de arhitectură de sistem (inclusiv descrierea modelelor în limbaj UML, care va include destul de multe detalii despre arhitectura sistemului)	Scrum Master/Team Lead and Development Team
2	Backlogul Produsului	Product Owner Scrum Master/Team Lead
3	Planul de Lansare	Product Owner Scrum Master/Team Lead
4	Strategia de testare, teste de acceptanță de către utilizatori, teste de încărcare și de stres și planuri de automatizare	QA/Software Tester
5	Planul de mentenanță	Scrum Master/Team Lead

7.9 DevOps

Odată cu adoptarea **DevOps**, software-ul trece rapid de la testarea și stadializarea dezvoltării. Mediul care găzduiește aceste aplicații este furnizat rapid, funcționând deseori într-un serviciu cloud.

Echipele de dezvoltare trebuie să proiecteze, să dezvolte, să livreze și să execute software-ul cât mai rapid și în cel mai sigur mod posibil. DevOps-ul trebuie să identifice și să rezolve problemele cât mai curând prin monitorizarea, prezicerea eșecului, gestionarea mediului și remedierea problemelor. Combinând această abordare comună în DevOps cu capacitatea de a monitoriza și analiza blocajele și de a optimiza cât mai rapid.

În multe cazuri, echipa noastră de proiect consideră viabilă pentru un proiect/produs implementarea unei practici DevOps care poate să adauge o valoare organizației dvs. printr-o serie de beneficii. DevOps-ul va acoperi o serie de procese din ciclul de viață al dezvoltării software:

- **Definiții și planificări**, care se concentrează pe planificarea fluxurilor de lucru DevOps pentru iterații, gestionarea versiunilor și urmărirea problemelor.
- **Scrierea codului, construirea și configurare**, care se concentrează pe dezvoltarea și revizuirea codului, gestionarea codului sursă și combinarea codului
- **Testarea** – verificarea dacă calitatea software-ului lansat și a codului sunt menținute pe tot parcursul procesului de dezvoltare și că codul de cea mai înaltă calitate este lansat în producție.
- **Pre-producție** se referă la activitățile implicate odată ce lansarea este pregătită pentru implementare; se mai numește punere în scenă.

- **Lansarea, implementarea și orchestrarea**, care este procesul de eliberare a software-ului și implică de obicei gestionarea schimbărilor, aprobările de eliberare, automatizarea lansării, orchestrarea programului, aprovizionarea și implementarea în producție.
- **Managementul continuu și configurația** include automatizarea configurației continue, gestionarea configurației și infrastructura ca cod.
- **Monitorizarea performanței aplicației** - ajută la identificarea problemelor care pot afecta experiența de utilizare.

Echipa noastră Scrum va produce software în iterații scurte în conformitate cu un program de livrare continuă de noi funcții și remedieri de erori în cicluri rapide de la două până la patru săptămâni. În schimb, DevOps va reuni echipele de dezvoltare și operații pentru a se concentra pe eliminarea silozurilor pentru a reduce timpul de adresare a feedback-ului clienților și va descompune blocajele pentru a permite livrarea continuă a software-ului. În consecință, echipa poate construi, testa și elibera software mai rapid cu cât mai multă eficiență și viteză.

Operațiunile noastre DevOps vor include următoarele:

- Integrarea continuă - codificarea, construirea, integrarea și testarea.
- Livrarea continuă - integrarea continuă, dar care se concentrează în principal pe lansările de produse.
- Desfășurarea continuă - automatizarea lansărilor proiectelor cât mai curând posibil.
- Efectuarea operațiunilor de dezvoltare a gestiunii configurațiilor și monitorizării continue.

7.9.1 Livrabile

#	Livrabile	Responsabil
1	• Ghid de integrare API; • Eșantioane de integrare în .NET; Java, PHP, Python • Descrierea umană și cea prelucrabilă automat într-un limbaj de descriere standard (de exemplu, WSDL sau Swagger).	Scrum Master/Team Lead DevOps Specialist Integration Specialist
2	Furnizarea codului sursă pentru componentele sistemului	Scrum Master/Team Lead DevOps Specialist
3	Implementarea sistemului și actualizările sistemului	Scrum Master/Team Lead DevOps Specialist

7.10 Managementul Lansărilor

Managementul lansărilor cuprinde planificarea, coordonarea și verificarea implementării soluțiilor IT în producție. Managementul lansărilor necesită colaborarea echipei de proiect care produce soluția și persoanele responsabile pentru infrastructura IT operațională a organizației dvs.

Scopul managementului lansărilor este de a coordona dezvoltarea, operațiunile și desfășurarea software-ului, asigurând în același timp alinierea la prioritățile de afaceri. Procesul este construit pentru mai multe obiective cheie:

- Gestionarea riscului;
- Coordonarea resurselor IT;
- Asigurarea proceselor de conformitate și audit;
- Supravegherea cutover-ului către versiuni noi;
- Asigurarea alinierii afacerii cu dezvoltarea de software.

7.10.1 Componentele administrării proceselor

- **Pipeline-ul pentru release** - un proces de lansare specific de la planificarea funcțiilor până la livrare;
- **Fluxul valorilor de eliberare** - procesele de eliberare care adaugă sau creează valoare pe rețea de lansare;
- **Politica de lansare** - definiția tipurilor, standardelor, cerințelor de guvernare pentru o organizație;
- **Modelul de lansare** - un singur proces de repetare a fluxului de lucru pentru conducta de eliberare care include activități umane și automatizate și respectă politicile de eliberare ale unei organizații;
- **Planul de lansare** - o instanță a unui șablon de lansare dezvoltat pentru o versiune specifică;
- **Planul de implementare** - activități pentru implementarea unei versiuni în mediul de producție;
- **Unitatea de lansare** - setul de artefacte dezvoltate pentru a implementa o caracteristică specifică;
- **Pachetul de lansare** - o combinație a uneia sau a mai multor unități de lansare implementate împreună ca o singură versiune din cauza interdependențelor, a planificării sau a priorităților de afaceri;
- **Lansările majore** - pachete de lansare ocazionale care includ adesea multe unități de lansare care au un impact important sau critic asupra afacerii;
- **Lansările minore** - pachete de lansare mai frecvente cu mai puține unități de lansare care nu includ componente critice pentru misiune.

Factorii de proces care vor fi luați în considerare pentru gestionare a versiunilor și Notele de Versiune sunt:

1. **Planificarea lansării** - Product Owner cu Scrum Master va identifica și va crea programul de lansare care va conține Note de lansare pentru fiecare Sprint planificat.
2. **Gestiunea configurației infrastructurii**. Echipa de proiect va colabora strâns cu Product Owner, pentru a realiza gestionarea configurației mediului operațional. Pentru a lansa în siguranță în producție, trebuie de știut ce este în prezent în producție și modul în care acele elemente hardware și software depind unul de celălalt. Cu cât este mai complexă infrastructura operațională și cu cât sunt mai multe echipe de dezvoltare, cu atât acest factor de proces devine mai important.
3. **Pregătirea pentru producție.** O parte a procesului de lansare este de a verifica dacă soluția este gata de a fi implementată și că părțile interesate sunt gata să o implementeze. Cu cât versiunile sunt mai mari și mai rare, cu atât aceasta devine mai mult o problemă.
4. **Echipele de livrare pentru sprijin.** Beneficiarul produsului, va colabora strâns cu echipa de proiect pentru a-i ajuta să-l implementeze cu succes.

7.10.2 Parcursul lansării

Planificarea lansării:

- Definirea cronologiei de livrare;
- Definirea datelor de livrare;
- Definirea cerințelor.

Dezvoltare propriu zisă

Odată cu finalizarea planului de lansare, vom începe să proiectăm și să construim produsul pentru lansare. Aceasta este „dezvoltarea” propriu zisă a produsului pe baza cerințelor expuse în planul de lansare.

Odată ce toate problemele care au apărut sunt abordate, este timpul să supunem construcția testelor de scenarii din lumea reală.

Acest lucru ar putea implica mai multe iterații. Pe măsură ce echipa creează produsul, acesta este trimis (de obicei automat) într-un mediu de testare pentru acceptarea utilizatorului. Aceasta permite echipei să identifice eventualele erori sau probleme care pot apărea într-un mediu real.

Pe măsură ce problemele sunt identificate, build-ul este trimis înapoi pentru dezvoltare în etapa a doua. Cu alte cuvinte, în cadrul procesului iterativ de gestionare a lansării, lucrul poate trece de la etapa a doua la etapa a treia și înapoi până la aprobarea lansării.

Testarea acceptării utilizatorului

Testarea de acceptare a utilizatorilor, cunoscută și sub denumirea de UAT, este atunci când utilizatorii finali navighează prin produs și oferă feedback. Acest lucru se realizează adesea sub forma unei versiuni beta gratuite online sau distribuite unui grup mai mare de angajați din cadrul companiei.

Testarea de acceptare a utilizatorilor este cel mai crucial pas pentru gestionarea a versiunilor, datorită cantității de date colectate și a corecțiilor necesare pentru a aduce build-ul la nivelul necesar pentru lansarea oficială.

Așa cum am menționat anterior, aceasta face parte dintr-un proces iterativ. Pe măsură ce bug-urile sunt identificate, echipa se întoarce la planșetă pentru a remedia problemele și a reproiecta build-ul pentru o mai mare integritate. Build-ul trebuie să treacă de etapa UAT pentru a fi luată în considerare pentru implementarea finală și lansare.

Pregătirea lansării

Acest pas constă în punerea finală a produsului, ținând cont de tot ceea ce a fost descoperit în UAT. Pregătirea versiunii include, de asemenea, o analiză finală a calității de către Testerul Software.

În cursul revizuirii, testerul software va efectua verificări finale pentru a se asigura că build-ul respectă standardele minime acceptabile și cerințele de afaceri prezentate în planul de lansare.

Deși UAT și asigurarea calității nu pot întotdeauna să reproducă fiecare scenariu care ar putea apărea odată ce produsul este lansat, acești pași speră ca cele mai comune erori, astfel încât echipa ta să poată anticipa și să prevină mai bine orice probleme la lansare.

După finalizarea revizuirii, echipa funcțională va valida constatările și va finaliza comunicarea pentru implementare. Înainte ca build-ul să poată fi implementată într-un mediu viu, aceasta trebuie să fie aprobată de Beneficiarul produsului.

Declanșarea lansării

Ziua cea mare a sosit în sfârșit și iată că munca grea a întregii echipe este răsplătită. Este timpul să lansăm produsul pe mediul de producție.

Pe lângă simpla trimitere a acumulării în producție, etapa de implementare include, de asemenea, mesageria și educația produsului atât utilizatorului final, cât și companiei dvs. în general.

De exemplu, utilizatorii ar trebui să fie înștiințați cu privire la modificările cu versiunea și cum să funcționeze în noile funcții. În funcție de cât de importante au fost schimbările, poate fi necesar să oferiți o pregătire robustă și continuă pentru ca toată lumea să fie la curent.

Acest lucru este important în special pentru versiunile interne în care angajații care folosesc software-ul trebuie să înțeleagă pentru a-și face munca eficient și productiv.

În cele din urmă, în etapa de desfășurare, echipa de proiect ar trebui să se întâlnească pentru a evalua performanța lansării și pentru a discuta despre modul în care a fost

desfășurată. În cazul în care există probleme persistente, acestea ar trebui identificate și documentate pentru ca echipa să se ocupe de ele în următoarea iterație.

7.11 Actualizarea backlog-ului de produs

- Product Owner-ul va menține produsul actualizat, astfel încât acesta să corespundă listei funcționalităților dorite;
- Răspuns la întrebări venite de la dezvoltatori - Product Owner va fi oricând la dispoziția echipei de dezvoltare pentru a răspunde eventualelor lor întrebări de clarificare, evitând astfel o comunicare complexă și formală în cadrul proiectului. Acest lucru este esențial pentru a vă asigura că echipa are toate informațiile la timp pentru a livra un produs de lucru la sfârșitul sprintului;
- Acceptarea pachetelor de lucru - pachetele de lucru livrate sunt prezentate Beneficiarului pentru a fi acceptate la sfârșitul fiecărui sprint. Beneficiarul acceptă pachetul de lucru sau notifică Furnizorul cu privire la orice defecte în timpul următorului sprint.

Deși nu este strict necesar, Beneficiarul produsului poate participa la ședințe de stand-up în echipă, ascultând progresele și eventualele blocante pentru o reacție imediată.

De asemenea, Beneficiarul produsului decide cu privire la versiunile de produs, conform planului de lansare. Conform principiilor metodologiei de gestionare a proiectului Agile, Beneficiarul va defini Declarația de viziune asupra produsului și Foaia de parcurs a produsului pentru a urmări progresul și pentru a asigura dezvoltarea corespunzătoare a produsului.

8. Iterația de tranziție

Aceasta este faza de finalizare a proiectului; toate cerințele stabilite au fost îndeplinite. În această fază, nu există mai multe elemente sau probleme (în produsul nerezolvat) care trebuie abordate și nu se pot identifica alte altele noi. Produsul este gata de livrare și această acțiune este pregătită (prin integrare, testare, documentare).

Faza de implementare și testare funcțională implică configurarea și integrarea componentelor sistemului pe infrastructura situată în mediul beneficiarului. Bazele de date, serverele de aplicații, modulele funcționale și alte componente vor fi configurate, realizând, de asemenea, interconectarea modulară. Ulterior, sistemul este testat din perspectivă funcțională, iar rezultatele testelor sunt înregistrate în rapoartele de testare. Această activitate are ca scop identificarea potențialelor erori ale programului, care ar putea avea un impact negativ asupra activității viitoare.

Faza de implementare este realizată cu perturbarea minimă a activității utilizatorilor. Aceasta poate include deplasarea utilizatorului, instalarea configurației hardware și software,

a aplicațiilor, precum și integrarea cu sistemele existente. Pentru a minimiza timpul de inactivitate al activității organizației, testele de sistem se efectuează pe configurații de test, situate în interiorul sau în afara instituției.

8.1 Testarea de acceptanță finală

Scopul testării de acceptare este de a confirma dacă sistemul funcționează în conformitate cu așteptările și cerințele. Testarea acceptării este centrată pe cerințele utilizatorilor (beneficiarilor).

Testarea acceptării are scopul de a confirma faptul că produsul livrat îndeplinește cerințele inițiale identificate în faza de evaluare. Pe parcursul acestui proces, Beneficiarul verifică dacă setul de specificații cerute, prevăzut în raportul de evaluare, se regăsește printre funcționalitățile oferite de produsul livrat.

Testarea acceptării are ca rezultat o matrice de conformitate a produsului cu specificațiile cerute. Implementarea și integrarea aplicațiilor este urmată de testarea funcțională, care presupune:

- Identificarea funcțiilor pe care trebuie să le îndeplinească sistemul (în corelație cu raportul de evaluare);
- Crearea datelor de intrare pe baza specificațiilor;
- Determinarea rezultatelor pe baza specificațiilor;
- Executarea cazului de testare;
- Compararea ieșirii sistemului cu puterea preconizată.

8.2 Testarea finală de sistem

Testarea nefuncțională, adică testarea caracteristicilor nefuncționale ale sistemului, are scopul de a măsura caracteristicile produsului care pot fi cuantificate la scară largă. Testarea nefuncțională include următoarele:

- **Testarea utilizabilității** - determină măsura în care produsul este înțeles, ușor de învățat, ușor de utilizat, care face apel la utilizatori;
- **Teste de stres** - evaluează sistemul dincolo de limitele specificate;
- **Testarea securității** - investigarea funcțiilor care detectează amenințările;
- **Testarea capacității de stocare** - analizează memoria ocupată și validarea capacității de stocare necesare;
- **Testarea performanței** - determină performanța sistemului;
- **Testarea recuperării** - testează opțiunile de rezervă și recuperare în caz de incidente;
- **Testarea volumului de date** - numai pentru sistemele care gestionează volume extreme de date;
- **Testarea instalării** - validează instalarea corectă a produsului;

- **Testarea documentației**;
- **Testarea încărcăturii** - comportamentul sistemului în condiții de încărcare sporite.

8.3 Instruirea utilizatorilor finali

Ședințele de instruire includ aspecte teoretice și/sau aspecte practice privind demonstrația pentru livrarea eficientă a transferului de competențe, metodologia la fața locului de muncă este selectată, deoarece permite participanților să învețe prin efectuarea efectivă a unor sarcini specifice la locul de muncă sau prin simularea loc de muncă. Fiecare metodologie de instruire este prezentată în Curriculumul de instruire și va fi descrisă în faza de inițiere a instruirii.

Evaluarea participanților va avea o abordare informală, formatorul va evalua imediat participanții cu privire la performanță, în timp ce va îndeplini sarcinile atribuite în timpul instruirilor la locul de muncă.

Limba de instruire - ședințele de instruire se desfășoară în limba română sau engleză, după caz. Limba sesiunii de instruire este definită în timpul procedurilor de inițiere și organizare a instruirii descrise mai jos.

Documentație de instruire - programe de instruire, cursuri de instruire (manuale, tutoriale video, teste etc.) pentru administratori, furnizori de servicii și utilizatori finali (persoane fizice și juridice) dezvoltate în platforma de învățare electronică.

Instruirea utilizatorilor sistemului este realizată diferit, în funcție de competențele fiecărei categorii de utilizatori, de preferință în funcție de grupurile de utilizatori.

Procesul de instruire se desfășoară diferit, în funcție de nivelul de cunoștințe al fiecărui grup de utilizatori, în conformitate cu metodologiile standard de pregătire în materie. Consultantul va oferi sesiuni de instruire on-line utilizând module de învățare electronică dezvoltate bazate pe Moodle LMS.

Rapoarte de instruire, prezentate după fiecare sesiune de formare, inclusiv:

- Lista participanților;
- Agenda sesiunii de instruire;
- Materiale de instruire (prezentări, laboratoare etc.);
- Rezultatele testelor stagiarilor.

De asemenea, echipa de proiect va informa Beneficiarul proiectului cu privire la orice cerințe, cum ar fi:

- **Cerințe de echipament** - identificarea resurselor materiale necesare susținerii sesiunii de instruire, cum ar fi, dar fără a se limita la hardware, software, rețea etc.
- **Cerințe de mediu**, cum ar fi, dar fără a se limita la: condiții, cerințe de instalații, locație etc.

- **Cerințe de personal** - identificarea resurselor umane necesare pentru susținerea programului sesiunii de formare, inclusiv a formatorilor.
- **Dependențe și limitări**, dacă există.

9. Lansarea în producție

În această fază, produsul livrat a fost validat și acceptat de către beneficiar, iar utilizatorii au fost instruiți cu privire la utilizarea sistemului.

Eventualele erori sau defecțiuni identificate după acest moment constituie obiectul contractului de garanție, iar dezvoltarea funcționalităților suplimentare necesită încheierea de acorduri cu furnizorul.

Data de implementare a sistemului este data la care termenul de garanție începe să se desfășoare.

10. Raportare de proiect

Conform metodologiilor de lucru descrise în această ofertă tehnică, au fost stabilite mecanisme de comunicare pentru a se asigura că informațiile necesare sunt generate și utilizate într-un mod eficient. În acest context:

- **Analiza progresului** este utilă pentru a revizui progresul comparativ cu planul. Aceasta poate fi, de asemenea, o oportunitate pentru prezentarea și discutarea rapoartelor scrise sau pentru o evaluare orală a problemelor actuale;
- **Rapoartele de progres** ale proiectului furnizează un rezumat al progresului proiectului, inclusiv informații cheie din indicatorii fizici și financiari, inclusiv în logica și graficele de activitate și în graficul resurselor.

Rapoartele de progres vor fi redactate într-un format standard care să permită compararea rapoartelor în timp. Scopul rapoartelor de progres va fi furnizarea de actualizări ale performanței în comparație cu reperele și reperele.

Echipa de proiect va pregăti rapoarte pe toată perioada contractului. Rapoartele vor acoperi toate activitățile proiectului și vor puncta toate rezultatele obținute de Beneficiarul produsului.

Raport de progres pentru Sprint

Aceste rapoarte vor prezenta progresul principal al perioadei raportate, evoluția activităților și întârzierilor, dacă acestea sunt semnificative, dificultățile întâmpinate, abaterile de la planul de activitate și consumul de resurse. Fiecare raport de activitate va fi însoțit de un raport financiar.

Beneficiarul produsului i se va furniza, la solicitare, rapoarte de progres intermediare pe probleme specifice identificate de reprezentanții lor.

Rapoartele vor furniza, de asemenea, informații despre rezultatele fiecărei etape, soluțiile date și deciziile majore care trebuie luate în considerare la luarea în considerare a aspectelor concrete ale activității.

Raport final

Versiunea preliminară a Raportului va fi transmisă Beneficiarului Produsului cu cel puțin o lună înainte de sfârșitul perioadei de execuție a contractului pentru analiză. Acesta va descrie întregul proces de implementare și va facilita evaluarea rezultatelor obținute atât din punct de vedere calitativ cât și cantitativ.

Raportul final va include:

- evaluare a succesului și a constrângerilor majore pentru fiecare activitate și sarcină;
- realizările generale ale contractului;
- evaluarea realizării rezultatelor propuse în contract;
- recomandări pentru acțiunile viitoare, în vederea asigurării durabilității activităților, a rezultatelor așteptate după finalizarea contractului și a măsurilor care trebuie luate de către Beneficiarul produsului în acest sens.