Az elsőrangú Product Owner készségei

> » A Product Owner felelős a Scrum Csapat munkájának eredményeként előállt termék értékének maximalizálásáért. Ennek megvalósítása a szervezeti formától, a Scrum Csapatoktól és egyénektől függően eltérő lehet. «

A Product Owner az a személy akinek egy jól körülhatárolható elképzelése van a termék víziójáról. Nem ismeri a jövőbeni termék működésének minden részletét, nem tudja pontosan hogyan fog működni a termék, de nagyon konkrét elképzelése van arról, hogy **miért** akarja a terméket fejleszteni és hogy **milyen** problémákat fog megoldani a termék és **kinek** szánjuk.

A fejlesztési folyamat fontos szereplői a stakeholderek. Ők azok akik használni, támogatni fogják a terméket vagy bármilyen formában hatással lesz az életükre vagy a munkájukra.

A stakeholderek igényei user storyk formájában jelennek meg a csapat asztalán. Például egy online áruház esetében a fizetés lehet egy ilyen story. A storyk leggyakrabban a termék egy funkcióját vagy a felhasználó egy lehetséges útját írják le. A stakeholdereknek rengeteg ötlete van, a Product Owner feladata, hogy ezeket egymástól független user storykká alakítsa át.

Valakinek meg is kell építenie a vízióban elképzelt terméket, akik pedig nem mások mint a Fejlesztők. A Fejlesztők kis méretű, keresztfunkcionális és önszerveződő csapatban, dolgoznak együtt - ideális esetben fizikailag is egy helyen.

Mivel agilis működésről beszélünk ezért a Fejlesztők nem egy nagy “Big Bang” indulásban gondolkodnak a termékkel kapcsolatban, hanem minél hamarabb szeretnék a felhasználókhoz eljuttatni a terméket és minél gyakrabban szeretnének új funkciókat hozzáadni. Átlagosan 4-6 user storyt tudnak fejleszteni egy Sprintben.

Vannak nagyobb sztorik amik kétszer akkorák és vannak kisebbek amik csak fele akkorák, de az átlag 4-6 körül mozog.

Annak érdekében, hogy a csapat ezt a mennyiséget folyamatosan tudja hozni komoly hangsúlyt helyeznek az automatikus tesztelésre és a folyamatos integrációra.

A probléma abban rejtőzik, hogy a stakeholdereknek rengeteg ötlete és javaslata van. Egy-egy Sprintben jóval több mint 4-6. Sőt minél több dolgot látnak működés közben, annál nagyobb ihletet kapnak újabb és újabb igények megfogalmazására.

Mi történik, ha ezeket mind egyszerre akarják a Fejlesztők megvalósítani. A csapatot maga alá fogja temetni a munka. Tegyük fel, hogy egy Sprintben 10 feladatot vállal magára a csapat, de csak 4-6-ot tud megoldani. A túlterheltség hatására megpróbálnak egyszerre több feladaton dolgozni, elvesztik a fókuszt ami demotiváltsághoz és a minőség csökkenéséhez fog vezetni. Végül egy vesztes-vesztes szituációban kötünk ki.

Ez ahhoz hasonló helyzet mintha több papírt próbálnánk a nyomtatóba tenni annak érdekében, hogy gyorsabban nyomtasson. Ha figyelmen kívül hagyjuk azt az alapvetést, hogy a bemenet nagyobb a kimenetnél akkor az elkerülhetetlenül elakadáshoz vagy szivárgáshoz fog vezetni.

A Scrum ennek elkerülésére a “tegnapi időjárás” módszert használja. A csapat megnézi mennyi volt az előző Sprint kimenete, legyen ez mondjuk 4-6 user story. Ezek után megnézik melyik az a 4-6 user story amivel a következő sprintben tudnak foglalkozni. A Product Owner feladata, hogy a lehetséges összes user story közül kiválassza azt a 4-6-ot, amit a következő Sprintben a legfontosabbnak tart a stakeholderek szemszögéből.

A Kanban megoldása, hogy határt szab a párhuzamosan végezhető feladatoknak. Tegyük fel, hogy a csapat egységesen 5 feladaton tud dolgozni egyszerre úgy, hogy mindenkinek jut elegendő feladat és nem terhelődik túl a csapat. Ezért úgy döntenek, hogy 5 story lesz az egyszerre végezhető feladatok maximuma. Csak akkor kezdenek neki új story megvalósításának, amikor egyet már befejeztek, így garantálva azt, hogy folyamatosan egységes mennyiségű új funkciót szállítanak.

Mindkét megoldás működhet jól. Mindkét megoldás esetén felhalmozódik jónéhány igény, ami várja, hogy sorra kerüljön egy fejlesztési ciklusban. A Scrum ezeket hívja Termék Backlognak.

> » A Product Owner felelős a Termék Backlog hatékony menedzseléséért is, beleértve:
>
> - A Product Goal meghatározását és egyértelmű kommunikálását
> - A Product Backlog elemeinek elkészítését és világos kommunikálását
> - A Product Backlog elemeinek sorrendezését; valamint
> - Annak biztosítását, hogy a Product Backlog könnyen áttekinthető, látható és mindenki számára érthető legyen «

Ezt a sort valakinek karban kell tartani. Ha Sprintenként 10 új igény érkezik be a stakeholderektől és a Fejlesztők csak 4-6 új funkciót tudnak szállítani, akkor elég gyorsan eljuthatunk oda, hogy egy-egy új elem akár 6 hónapig is várhat míg sorra kerül. Így kialakulhat az a helyzet, hogy olyan dolgot fejleszt a csapat, amit valaki 6 hónappal korábban kért. Ez nem túl agilis.

Csak egy módon lehet kordában tartani az egyre növekvő számú igények sorát. Ez pedig nem más, mint a **NEM** szó. Ez a legfontosabb szó egy Product Owner szókincsében. Érdemes is minden nap a tükör előtt gyakorolni. Egy új kérésre igent mondani rendkívül könnyű. A legnehezebb feladat egy Product Owner munkájában

az, hogy eldöntse mi lesz az a funkció ami nem fog bekerülni a termékbe és természetesen vállalja a felelősséget az ezzel járó következményekért.

A Product Owner feladata, hogy eldöntse mi az amin a fejlesztőcsapat elkezd dolgozni és mi az a minőség ami a fejlesztési folyamat végén átadható a stakeholdereknek. Szintén az ő feladata az, hogy eldöntse mi az ami a következő iteráció során kerül be a fejlesztési folyamatba és mi az ami csak később. Nem könnyű feladat, és hatékonyan csak a Fejlesztők és a stakeholderek aktív bevonásával kivitelezhető.

Ahhoz, hogy a Product Owner megfelelő sorrendet tudjon felállítani az egyes elemek között ismernie kell, hogy egy-egy elem mekkora üzleti értéket képvisel és mekkora erőfeszítést igényel a fejlesztése. Néhány funkció kritikus a működés szempontjából, néhány pedig inkább csak hab a tortán. Néhány órák alatt elkészül másokhoz akár hónapok is szükségesek.

Mi lehet a különbség egy-egy user story értéke és mérete között? Így van - semmi! Az, hogy egy user story-t nagy energiabefektetéssel lehet csak megvalósítani, nem jelenti azt, hogy üzletileg ugyanolyan nagy értéket képvisel. Gondoljunk bele, hogy egy olyan egyszerű funkció, mint az automatikus mentés egy szövegszerkesztőben, mennyire fontos lehet, mekkora értéket képvisel a felhasználóknak és ezek mellett mennyire alacsony a fejlesztési költsége. Ugyanakkor a Microsoft Office gemkapocs funkciója jelenetős energiákat emészthetett fel a fejlesztés terén, míg fogadtatása és haszna nem egészen azt mutatta, hogy a felhasználók túlzottan lelkesedtek érte.

Az egyes user storyk értéke és mérete azok a tulajdonságok amik, segítenek a Product Ownernek, hogy kellően megalapozott döntést hozzon amikor sorba rendezi őket.

Ha két azonos méretű user story között kell eldönteni a sorrendet, akkor a nagyobb értékű fog előre sorolni, míg két üzletileg azonos értékű közül a kisebb energia ráfordítást igénylő kerül előrébb a sorban.

Itt merülhet fel a jogos kérdés, honnan tudja a Product Owner az egyes user storyk méretét vagy értékét. A rossz hír az, hogy ezek nem határozhatóak meg egyértelműen. Egyszerűen a megérzéseire kell hallgatnia. A későbbiek során megismerkedünk majd olyan technikákkal amelyek a megérzéseknél egy fokkal jobban segítenek az egyes storyk értékének rangsorolásában.

A megérzésein túl be kell vonnia a stakeholdereket is, hogy folyamatosan felmérje, hogy az adott pillanatban mi az amit értéknek tartanak. Az egyes user storyk méretének becsléséhez pedig folyamatosan beszélnie kell a Fejlesztőkkel, hogy ők is visszajelzést adjanak a fejlesztésre fordítandó energiáról. Az összes ilyen becslés csak relatív és nem abszolút. Nem tudom megmondani segédeszköz nélkül, hogy mekkora egy alma és egy eper pontos súlya, de azt elég biztosan meg tudom állapítani, hogy az alma ötször nagyobb mint az eper és az eper nekem jobban ízlik. Ennyi elég is ahhoz egy Product Ownernek, hogy sorba tudja állítani a user storykat.

Egy új projekt indulásakor szinte az összes becslés nagyot fog tévedni, de ez teljesen rendben is van így. Ami számít ilyenkor az a becslések során elkezdődő párbeszédek és nem a becslés tényleges végeredménye. Minden egyes alkalommal amikor a csapat szállít valami kézzel foghatót a felhasználóknak visszajelzést kapnak. Ezek a visszajelzések pedig egyre javítják a becsléseket mind méret, mind érték szempontjából. Ezért szükséges folyamatosan sorba rendezni a user storykat és újra megbecsülni azokat. A teljes víziót megbecsülni az elején elég rossz hozzáállás, hiszen a projekt elején áll rendelkezésünkre a legkevesebb információ. A visszajelzések a Product Owner legjobb barátai.

A user storyk sorba rendezése önmagában nem elég. Ahhoz, hogy a termék korán a felhasználóknál legyen és gyakran tudjuk frissíteni a meglévő user storykat apró falatnyi részekre kell bontani. Ideális esetben egy ilyen falat nem igényel néhány napnál több munkát. Egy tölcsérbe kell töltenünk a user storykat ahol a tölcsér szájánál kicsi és jól definiált elemek találhatóak, míg a bemenetnél a sokkal tágabban értelmezett és nagyobb elemek. Ha ezt a lebontást mindig akkor csináljuk meg, amikor az adott funkció közel kerül a fejlesztéshez, azzal tudjuk azt garantálni, hogy mindig a lehető legaktuálisabb információk alapján hozunk döntést az egyes funkciókról.

A fentieket - tehát az egyes user storyk méretének és értékének becslését, sorba rendezését és lebontását - összességében backlog refinementnek nevezzük. A Product Owner heti egyszer összehívja a Fejlesztőket - esetleg csatlakozik néhány stakeholder is - és egy műhelymunka keretében egyeztetnek a backlogról. A napirend változhat, néha a becslés, néha a user storyk lebontása a cél.

Az eddigiekből kirajzolódni látszik egy mintázat: kommunikáció! A Product Owner legfontosabb feladata a folyamatos kommunikáció. Ez teljesen összhangban áll az Agilis Kiáltvány első pontjával:

> **“Az egyéneket és a személyes kommunikációt (értékeljük)** a módszertanokkal és eszközökkel szemben”

A Product Owner feladata tehát nem az, hogy kiskanállal etesse a csapatot user storykkal. Ez egyrészről unalmas másrészről nem túl hatékony. A Product Owner feladata, hogy mindenkivel megértesse mi a víziója, biztosítsa a közvetlen kapcsolatot a Fejlesztők és a stakeholderek között és hogy elősegítse, hogy a valódi felhasználóktól érkező visszajelzések minél hamarabb beépülhessenek a termékbe. Ezzel segítve a Fejlesztőket abban, hogy egyre önállóbban tudjanak szakmai döntéseket hozni, míg a Product Owner a teljes képre fókuszál.

A Product Ownernek és a Fejlesztőknek jó pár kompromisszumos döntést kell meghozni a munkájuk során. Nézzünk néhány példát.

Elsőként kompromisszumot kell kötni a különböző értékek között. Egy új projekt elején a bizonytalanság és a kockázatok jelentik a legnagyobb veszélyt.

Létezik üzleti kockázat: biztos, hogy a megfelelő dolgot fejlesztjük? Csoport kockázat: a Fejlesztő csapat összetétele lehetővé teszi-e, hogy megvalósítsuk a terméket? Technológiai kockázat: a platform amin szeretnénk megvalósítani megfelelő lesz a terméknek? Mennyire lesz skálázható? Létezik még ezen túl a költségek és a határidő kockázata is: Reális idő- és költség korlátok között be tudjuk fejezni a fejlesztést?

A tudás a kockázat ellentéte. Amikor tehát nagy a bizonytalansági tényező akkor a tudás megszerzésére és rendszerezésére kell fókuszálni - ilyen például egy felhasználói felület prototípusának elkészítése, technológiák kipróbálása vagy egyszerű kísérletezés. Ezek nem túl érdekesek a termék felhasználói oldaláról, de ettől még értéket hordoznak, hiszen csökkentik a jövőbeni kockázatot.

A jövőben a bizonytalanság fokozatos csökkenésével a csapat folyamatosan egyre többet tud az ügyfelek számára is értékes dolgokra koncentrálni. Egyre inkább kikristályosodik, hogy mit és hogyan szeretnénk fejleszteni és már csak a tényleges munka marad. Az értékesebb user storykkal kezdve pedig egy szép meredek értéknövekedést tud megvalósítani a csapat.

Az idő előrehaladtával ez a görbe ellaposodik. A legfontosabb funkciókat már használják az ügyfelek, a csapat már a bónusz funkciókra koncentrál, ez a hab a tortán. A fejlesztési folyamatnak ez a szakasza a legkényelmesebb, mivel bármely ponton az tudja mondani a Product Owner és a Fejlesztők, hogy abbahagyják az adott funkcióval kapcsolatos fejlesztést és más fontosabb funkcióval folytatják tovább. Ez az üzleti agilitás.

Ezek alapján amikor értékről beszélünk akkor a tudást és az ügyfél értéket értjük alatta és meg kell találnunk azt a kompromisszumot, ahol egyensúlyban van a kettő.

A második fontos kompromisszum a hosszú és rövid távú gondolkodás között lévő szakadékban rejlik. Mit fejlesszünk következőként? Előrébb vegyük-e a sürgős hibajavítást vagy fejlesszük inkább azt az új funkciót ami elkápráztatja az ügyfeleket, esetleg fejlesszük tovább a teszt rendszerünket ami majd gyorsabb fejlesztést tesz lehetővé a jövőben? Folyamatosan egyensúlyozni kell a reaktív és proaktív munka valamint a tűzoltás és a tűzvédelem között.

Ez egy másik kompromisszumhoz is kapcsolódik. Arra kell-e a hangsúlyt fektetnünk, hogy

“a megfelelő dolgot fejlesszük”

“megfelelő módon fejlesszük azt a dolgot” vagy

“gyorsan fejlesszük a dolgot”?

Ideális esetben mindhármat szeretnénk, de nem könnyű megtalálni az egyensúlyi pontot. Vegyünk egy egyszerű példát: a lehető legjobb terméket szeretnénk megvalósítani a lehető legtökéletesebb architektúrával. Ha túl sok időt töltünk a tökéletesítéssel, akkor elmulaszthatjuk a megfelelő időpontot a piacra lépéshez és könnyen pénzügyi problémákba futhatunk.

Vagy vegyünk egy másik példát: már a prototípussal ki szeretnénk lépni a piacra. Ez rövid távon akár jó is lehet, de hosszú távon rendkívüli mennyiségű technikai problémát cipelünk magunkkal és a csapat értékteremtő képessége közel lesz a nullához.

Vagy építhetünk rekord idő alatt egy csodás katedrálist, ha kiderül, hogy az ügyfelek valójában egy lakókocsit szerettek volna.

A Scrum szerepek között létezik egy egészséges feszültség a fentiek kapcsán: a Product Owner szeretné a megfelelő dolgot a felhasználók kezeiben látni. A Fejlesztők szeretnék a leginkább megfelelő módon megvalósítani azt a dolgot. A Scrum Master pedig igyekszik a lehető legrövidebbre venni az utat amin megérkezik a visszajelzés a csapathoz.

A sebesség fontos tényező. Ha hamar kap visszajelzést a csapat az felgyorsítja a tanulást, így hamarabb kiderül, hogy mi a megfelelő termék és hogyan lehet megfelelően megvalósítani. Mindhárom tényező fontos ezért érdemes megtalálni az egészséges egyensúlyt közöttük.

A harmadik és utolsó kompromisszum pedig az új termék fejlesztése és a régi termék javítása között található. A termék backlog lehet egy megtévesztő fogalom, hiszen azt jelzi, hogy csak egy termék van. A projekt szintén lehet megtévesztő, hiszen azt jelzi, hogy a termék fejlesztése egyszer véget ér. Egy termék valójában soha nem lesz “kész”, mindig van karbantartási és javítási feladat egészen addig, amíg az adott termék elér az életciklusa végére és megszüntetjük.

Mi történik a régi termékkel, ha egy új fejlesztésébe kezdünk? Átadni a terméket egy másik csapatnak költséges és kockázatos lehet. Ami ennél sokkal gyakoribb, hogy ugyanaz a csapat tartja karban a régi terméket aki fejlesztette azt és ezzel párhuzamosan fejlesztik az újat is. Így a csapat backlogja sokkal inkább kapcsolódik magához a csapathoz, mint egy konkrét termékhez - összességében olyan elemeket fog tartalmazni amelyeket a Product Owner szeretne megvalósítani a jövőben és akár több termékből is találhatóak meg funkciók benne. A Product Owner feladata pedig, hogy megtalálja ezen elemek között az egyensúlyt.

Néha egy-egy stakeholder megkeresi a Product Ownert és megkérdezi a következőket:

“Mikor lesz kész amit várunk?”

“Mennyi fog elkészülni belőle karácsonyig?”

A Product Owner feladata ebben az esetben a várakozások kezelése. Ami még ennél is fontosabb: a várakozások megtartása a realitás medrében. Ezt azt jelenti, hogy egy Product Owner semmiképpen nem fog hazudni a stakeholdereknek. Ez nem is annyira könnyű feladat ha belegondolunk, de az agilitás kemény dió tud lenni néha.

Előrejelzést nem annyira nehéz készíteni, egészen addig amíg nem kell túlságosan precíznek lennie. Ha megmérjük a csapat sebességét vagy az összes csapat együttes sebességét fel tudjuk rajzolni a “Burnup” grafikont. Ez a grafikon kumuláltan mutatja meg, hogy mennyi az összes eddigi szállított user story.

Vegyük észre a különbséget: ez a grafikon a fejlesztési folyamat kimenetét mutatja, ami nem egyezik meg az ügyfél értékkel. Nem az a célunk, hogy minél nagyobb legyen a kimeneti mennyiség, hanem az hogy minél több értéket tudjunk előállítani a mennyiség arányában. A kevesebb néha több.

Vegyünk egy burnup grafikont és rajzoljunk bele egy optimista és egy pesszimista trendvonalat. Ehhez nyúlhatunk a statisztika eszköztárához vagy egyszerűen behúzhatjuk kézzel is. A két trendvonal közötti különbség mutatja meg mennyire bizonytalan az adott csapat sebessége. Ez idővel egyensúlyba kerül és a bizonytalansági tölcsérünk egyre szűkebb lesz.

Rendben, vissza a várakozások kezeléséhez!

Tegyük fel, hogy a stakeholderek azt kérdezik a Product Ownertől:

“Mikor lesz kész MINDEN amit szeretnénk?”

“Mikor érkezünk a következő mérföldkőhöz?”

Ezeknél a kérdéseknél a scope rögzített az idő pedig változó tényező. A Product Owner ránéz a burnup grafikonra és azt mondja:

“Körülbelül április-május környékén”.

Tegyük fel, hogy a stakeholderek azt kérdezik a Product Ownertől:

“Mi lesz kész karácsonyra?”

Ebben az esetben az idő rögzített míg a scope egy változó tényező. A trendvonalra újra ránézve meg tudjuk mondani:

“X funkció teljesen kész lesz, Y egy része is, de Z egyáltalán nem”.

A harmadik lehetséges kérdés:

“Meglesznek EZEK a funkciók karácsonyra?”

Ebben az esetben mind a scope mind az idő rögzített tényező. A legjobb válasz ilyen esetben a Product Ownertől:

“Nem, sajnos ez nem lesz kész addigra.”

majd közvetlenül utána

“Nagyjából EZT és EZT tudjuk karácsonyra szállítani”

vagy

“még ennyi időre van szükség.”

Általánosságban sokkal jobb a scope-ot módosítani mint a fejlesztésre szánt időt kiterjeszteni. Ha először a scope-ot vágjuk meg, később még mindig dönthetünk úgy hogy több időt hagyva fejlesszük a korábban kihagyott funkciókat. Fordítva nem

működik, hiszen ha a fejlesztésre szánt idő elmúlt akkor azt visszaforgatni már nem tudjuk. A Product Owner végső válasza tehát:

“Tudunk szállítani egy részt a termékből a határidőre és a többit későbbre halasztjuk, vagy most nem szállítunk semmit és a kért funkciót EKKORRA tudjuk szállítani? Melyik lehetőség a szimpatikusabb?”

Az ehhez kapcsolódó számítások viszonylag egyszerűek, a Product Owner pedig folyamatosan tudja frissíteni az előrejelzését.

Fontos kiemelni, hogy a Product Owner ilyen esetben empirikus adatokra alapozza az előrejelzést a vágyálmok helyett. Ezen túl teljesen őszinte a bizonytalansággal kapcsolatban. Ez egy nagyon egyenes módja annak, hogy a stakeholderekkel kommunikáljon, akik jó esetben díjazzák is ezt. Ha a szervezetre nem jellemző az egyenes és őszinte kommunikáció az agilitás sem fog egyszerűen gyökeret verni.

Egy dologra azonban oda kell figyelni. Ha a Fejlesztők túl sok technológiai adósságot halmoznak fel - például nem írnak jó teszteket vagy nem fejlesztik folyamatosan az architektúrát - akkor a sebességük időről-időre lassulni fog, a burnup grafikon pedig ellaposodik. Ez szinte lehetetlenné teszi a becslést, ezért a Fejlesztők felelőssége, hogy egyenletes sebességgel dolgozzanak a Product Ownernek pedig el kell kerülnie, hogy olyan nyomást gyakoroljon a csapatra, amivel rossz döntésekre kényszerítheti őket.

Mi a helyzet akkor, ha egy nagyobb projekten dolgozunk több csapattal? Egy 5 fős csapat helyett mondjuk 3 csapat dolgozik a projekten. Több Product Owner is van és mindegyiknek saját backlogja is a termék különböző komponenseivel.

Összességében ugyanezeket az elveket alkalmazhatjuk ebben az esetben is. Szükség lesz a csapat kapacitásának kezelésére, az egyeztetésre a stakeholderekkel, Product Ownerekre akik tudnak nemet mondani, a backlog refinementre.. A projekt

sebességét a csapatok kimeneteinek összessége fogja megadni és ezt is fel lehet használni előrejelzések készítéséhez. Vagy - ha ennek van értelme - a csapatok külön is készíthetnek előrejelzést.

Az ilyen több csapatos felállásokban azonban a Product Ownereknek van még egy felelősségük: egymással is beszélni kell. A csapatokat és azok backlogjait úgy kell kialakítani, hogy minimálisra csökkentsük a függőségeket. Biztosak lehetünk benne, függőségek mindig lesznek, ezért a Product Ownereknek egyeztetni kell arról, hogy megfelelő sorrendben kerüljenek a termék egyes komponensei a fejlesztési ciklusokba.