Examensarbete, Högskolan på Åland, Utbildningsprogrammet för Informationsteknik

Grunderna inom indiespelutveckling

Daniel Hannus, Gabriel Persson

2025:36

Datum för godkännande: 27.05.2025

Handledare: Jonas Waller

EXAMENSARBETE

Högskolan på Åland

Utbildningsprogram:	Informationsteknik
Författare:	Daniel Hannus, Gabriel Persson
Arbetets namn:	Grunderna inom indiespelutveckling
Handledare:	Jonas Waller

Abstrakt

I vårt examensarbete undersöker vi vad som krävs för att utveckla och lansera ett indiespel, med fokus på både tekniska och affärsmässiga aspekter. Arbetet utgår från frågeställningen: vilka delar behöver en indieutvecklare behärska för att skapa och lansera ett spel? Genom praktiskt arbete i Unreal Engine och teoretisk undersökning av marknadsstrategier analyserar vi processen från idé till lansering. Resultatet visar att bred kompetens, tydlig planering och tidig marknadsföring är avgörande. Slutsatsen är att spelutveckling i liten skala kräver helhetssyn och långsiktigt engagemang.

Nyckelord (sökord)

Spelutveckling, Unreal Engine, Perforce

Högskolans serienummer:	ISSN:	Språk:	Sidantal:
2025:36	1458-1531	Svenska	67

Inlämningsdatum:	Presentationsdatum:	Datum för godkännande:
07.05.2025	22.05.2025	27.05.2025

DEGREE THESIS

Åland University of Applied Sciences

Degree Programme:	Information Technology
Author:	Daniel Hannus, Gabriel Persson
Title:	Basics in Indie Game Development
Academic Supervisor:	Jonas Waller

Abstract

In our thesis, we explore what is required to develop and publish an indie game, focusing on both technical and business aspects. The work is based on the question: what areas must an indie developer master to create and release a game? Through practical work in Unreal Engine and a theoretical study of marketing strategies, we analyze the process from concept to launch. The results show that broad competence, clear planning, and early marketing are crucial. We conclude that small-scale game development demands a holistic perspective and long-term commitment.

Keywords

Game Development, Unreal Engine, Perforce

Serial number:	ISSN:	Language:	Number of pages:
2025:36	1458-1531	Swedish	67

Handed in:	Date of presentation:	Approved:
07.05.2025	22.05.2025	27.05.2025

INNEHÅLLSFÖRTECKNING

1. Inledning

1.1 Syfte

Syftet med detta arbete är att utforska och dokumentera de olika stegen i indiespelutveckling, från idé och planering till utveckling, lansering och marknadsföring. Arbetet syftar också till att undersöka vad som kännetecknar ett indiespel, både ur ett tekniskt och affärsmässigt perspektiv, samt hur olika lanseringsstrategier och affärsmodeller påverkar slutresultatet.

Vi analyserar såväl tekniska som organisatoriska aspekter av spelutveckling i Unreal Engine 5 (Epic Games, 1995), med särskilt fokus på hur versionshantering med Perforce (Perforce Software, Inc., 1995) kan underlätta samarbete i mindre team. Utöver det kommer vi att belysa hur artificiell intelligens används i spelutveckling, särskilt i form av datorstyrda karaktärer (NPC:er), och hur dessa påverkar spelupplevelsen.

Målet är att ge en tydlig och praktiskt användbar överblick över processen att skapa ett indiespel – från idé till färdig produkt – samtidigt som vi identifierar vanliga utmaningar och möjligheter för utvecklare. Genom att kombinera teoretisk forskning med praktiska exempel och fallstudier strävar vi efter att ge en djupgående förståelse för de många aspekterna av indiespelutveckling.

1.2 Problemformulering

Att påbörja ett indieprojekt inom spelutveckling innebär en rad komplexa utmaningar. Även om tillgången till moderna verktyg och spelmotorer har sänkt tröskeln för att skapa egna spel, kräver utvecklingsprocessen fortfarande både teknisk kompetens och strategisk planering.

En av de främsta svårigheterna för indieutvecklare är bristen på resurser – i form av finansiering, arbetskraft och tid. Dessa begränsningar påverkar allt från grafik och ljuddesign till testning och lansering. Dessutom står indieutvecklare ofta inför tekniska hinder, såsom prestandaoptimering, kodhantering och samarbete i mindre team utan etablerade arbetsflöden.

Att hantera versionshantering, till exempel via verktyg som Perforce, blir avgörande för att undvika konflikter och dataförlust.

Utöver de tekniska aspekterna kräver indiespelutveckling även förståelse för juridiska frågor som rör upphovsrätt och licenser. Att använda tredjepartsresurser utan tydliga rättigheter kan medföra juridiska konsekvenser. Därtill måste utvecklaren ofta själv ansvara för marknadsföring och lansering, vilket kan vara svårt utan tidigare erfarenhet eller ett etablerat nätverk.

Konkurrensen på marknaden är dessutom hård, och många spel riskerar att försvinna i mängden utan en tydlig nisch eller målgrupp. För att lyckas krävs därför inte bara ett tekniskt genomförbart spel, utan också ett välformulerat koncept och en strategi för hur spelet ska nå ut till spelare.

Detta examensarbete syftar till att identifiera och analysera dessa utmaningar samt föreslå lösningar och strategier för att skapa ett hållbart och genomförbart indieprojekt – från idé till färdig produkt (iXie, 2023).

1.3 Metod

För att förstå processen att utveckla ett indiespel i Unreal Engine har vi använt en kombination av praktisk inlärning och teoretiska studier.

Vi har lärt oss miljön genom att:

- Ta del av tutorials och kurser – Vi har studerat videoguider och interaktiva kurser för att förstå grunderna i Unreal Engine, versionshantering med Perforce samt spelutvecklingens arbetsflöde. (Perforce, n.d.); (Epic Games Developer, n.d.-h).
- Skapa egna demoprojekt – För att tillämpa kunskapen i praktiken har vi utvecklat prototyper där vi testat speldesign, kodstruktur och tillgångshantering (asset handling).
- Läsa online-trådar och forum – Vi har aktivt sökt efter lösningar på vanliga problem genom att läsa diskussioner från andra indieutvecklare på plattformar som Unreal Engine Forum, Reddit och Stack Overflow.

- Studera litteratur om spelutveckling – Vi har tagit del av artiklar, bloggar och tekniska dokumentationer för att förstå viktiga koncept inom speldesign, programmering och versionshantering.

Dessa metoder har gett oss både en praktisk och teoretisk förståelse av utvecklingsprocessen, vilket har hjälpt oss att identifiera och hantera de vanligaste utmaningarna vid indiespelutveckling.

1.4 Avgränsningar

Detta slutarbete fokuserar på processen att utveckla ett indiespel från idé till lansering, med särskilt fokus på Unreal Engine som utvecklingsverktyg. Vi undersöker de tekniska, organisatoriska och juridiska utmaningar som indieutvecklare kan stöta på och hur dessa kan hanteras effektivt.

Specifikt kommer vi att behandla:

- Idégenerering och konceptutveckling – Hur man formulerar en spelidé och strukturerar ett projekt.
- Teknisk utveckling i Unreal Engine – Grundläggande arbetsflöde, versionshantering (med Perforce), spelkod, optimering och integration av tillgångar (assets).
- Tillgångshantering och licenser – Hur man tolkar licensvillkor för att använda tredje parts innehåll lagligt.
- Marknadsföring – Effektiva marknadsföringsstrategier och vilka som passar bäst för ett indiespelprojekt.
- Early Access och lanseringsstrategier – Hur man avgör när spelet är redo för spelare.
- Projektledning och arbetsstruktur – Hur man effektivt organiserar och genomför utvecklingsprocessen.

Detta behandlas inte i detta examensarbete:

- Djupgående speldesign teori – Vi kommer att diskutera grundläggande designprinciper men inte fördjupa oss i detaljerade speldesignmodeller.
- Alternativa spelmotorer och verktyg – Fokus ligger på Unreal Engine, och vi kommer inte att göra detaljerade jämförelser med exempelvis Unity eller Godot. Vi kommer

inte heller att djupgående behandla alternativa versionshanteringsprogram som exempelvis Git och SVN.

- Avancerad nätverksprogrammering – Vi berör grunderna i multiplayer och versionshantering men går inte in på detaljerad implementering av nätverksprotokoll.
- Underhåll av spelet efter lansering – vi berör endast aspekterna upp till lansering.

Dessa avgränsningar görs för att arbetet ska ha en tydlig riktning och vara hanterbar inom den givna tidsramen. Målet är att ge en praktiskt användbar vägledning för indieutvecklare som vill skapa och lansera sitt första spel med Unreal Engine.

2. Förståelse av indiespelutveckling

2.1 Vad är ett indiespel?

Ett indiespel, eller "independent video game," är ett spel som utvecklas av enskilda personer eller små team som arbetar självständigt från större spelstudior, både finansiellt och kreativt. Denna frihet möjliggör experimenterande med okonventionella narrativ, estetik och spelmekaniker, vilket ofta resulterar i unika och minnesvärda spelupplevelser. (Pajkovic, 2023)

Begreppet "indie" är dock inte alltid tydligt definierat. Även spel med större budgetar eller som distribueras av stora företag kan ibland betraktas som indie, beroende på spelets koncept, stil och ursprung. Exempelvis började *Minecraft* (Mojang, 2009) som ett litet indieprojekt innan det köptes av Microsoft. Därför avgörs ett spels "indieness" ofta av en kombination av faktorer snarare än en strikt definition. (Pajkovic, 2023)

2.2 Jämförelse med AAA-spel

AAA-spel och indiespel skiljer sig markant åt i flera avseenden, särskilt när det gäller budget, teamstorlek och utvecklingsprocesser, se figur 1.

AAA-studios är stora och välrenommerade företag som skapar spel med höga produktionsvärden, avancerad grafik och den senaste tekniken. De har ofta stora budgetar, längre utvecklingstider och tillgång till omfattande resurser. Arbetet i en AAA-studio präglas av en mer strukturerad miljö med specialiserade team och tydliga roller, men det kan också innebära längre arbetsdagar, mindre kreativ frihet och en mer konkurrensutsatt arbetsplats.

Indie-studior, däremot, är mindre team som arbetar med begränsade budgetar och kortare utvecklingscykler. De har större kreativ frihet och möjlighet att experimentera med okonventionella idéer och spelmekaniker. Utvecklare i indie-studior har ofta en bredare roll och kan påverka spelets riktning mer direkt. Nackdelarna kan inkludera lägre ekonomisk stabilitet, mindre resurser och ett större behov av att hantera flera uppgifter samtidigt.

Characteristics	Indie	AAA	AA
Development Budget	A few thousand to $1 million.	Over $50 million, sometimes more.	Variable, typically between indie and AAA.
Marketing Push	Relies on word of mouth, social media.	High budgets, extensive marketing efforts.	Between indie and AAA, often less.
Team Size	Small teams, sometimes one person.	Large teams, hundreds of employees.	Usually below 50 people.
Dedicated Publisher	Often lacks a dedicated publisher.	Typically published by established companies.	Backed by a publisher with more creative freedom.
Graphics & Technology	Focus on innovation, less cutting edge.	Cutting-edge technology and graphics.	High-quality, not cutting edge.
Production Values	Smaller scale, emphasis on mechanics.	High production values, famous voice actors.	Good production values, less famous actors.
Franchising	May lead to sequels post-success.	Intended to become franchises post-success.	Likely to become franchises if successful.

Figur 1. Skillnaden mellan olika typer av spelstudior (Zolotarenko, 2024)

Medan AAA-spel dominerar marknaden med stora lanseringar och högkvalitativa produktioner, bidrar indiespel till innovation och diversitet inom spelbranschen genom att våga ta kreativa risker och introducera nya koncept och berättelser. (Yasha_eix, 2023)

2.3 Utmaningar och möjligheter

Indieutvecklare står inför en unik uppsättning utmaningar och möjligheter när de skapar spel. En av de största utmaningarna är begränsade resurser, både i form av budget och arbetskraft. Utan stöd från stora studior måste indieutvecklare ofta hantera flera roller samtidigt, från programmering och grafik till marknadsföring och projektledning. Den mindre ekonomiska stabiliteten kan också innebära osäkerhet kring lön och finansiering av projektet.

Trots dessa hinder finns det många möjligheter för indieutvecklare. Den kreativa friheten är en av de största fördelarna, vilket gör det möjligt att utforska okonventionella spelidéer och ta risker som större studior kanske undviker. Indieutvecklare har också större flexibilitet i sina arbetsmetoder och kan snabbt anpassa sig till förändringar i projektet.

För att lyckas behöver indieutvecklare noggrant planera sin budget, effektivt hantera sin tid och vara strategiska i hur de marknadsför och distribuerar sina spel. Att bygga en stark community och ta emot feedback från spelare kan också vara avgörande för att skapa ett spel som når sin fulla potential. (Yasha_eix, 2023)

3. Förberedelser och planering

3.1 Idé och konceptualisering

Att ta fram en originell och genomförbar spelidé är ofta en av de mest krävande, men också mest kreativa och givande delarna av spelutvecklingsprocessen. En stark idé fungerar som grund för såväl speldesign, berättelse och estetik som teknisk struktur. Det finns flera etablerade metoder och angreppssätt för att stimulera denna typ av kreativitet (se även figur 2).

Inspiration från befintliga spel.

Analys av redan existerande spel – både framgångsrika och mindre lyckade – kan erbjuda insikter kring vad som fungerar väl i praktiken. Genom att studera vilka spelmekaniker, narrativa grepp eller visuella uttryck som varit särskilt engagerande, kan utvecklingsteam identifiera potentiella förbättringsområden eller vidareutveckla intressanta koncept som tidigare inte utforskats till fullo.

Genre- eller karaktärsdriven design.

Ett vanligt angreppssätt i tidiga idéfaser är att utgå från en etablerad genre, exempelvis plattformsspel, taktiska rollspel eller realtidsstrategi. Genom att ställa frågor som *"Hur kan denna genre förnyas?"* eller *"Vilka spelmekaniska element saknas inom genren?"* kan utvecklare identifiera utrymme för innovation.

Ett alternativt grepp är att utgå från en specifik karaktärsidé. I detta fall formas spelets mekanik och narrativ kring karaktärens egenskaper, bakgrund och mål. En intressant protagonist med unika färdigheter kan i sig motivera både spelvärldens utformning och spelupplevelsen som helhet.

Kreativa tekniker och verktyg.

Flera strukturerade metoder kan användas för att främja idégenerering:

- **Tankekartor (Mind mapping):** En visuell teknik där idéer förgrenas från en central kärntanke och vidareutvecklas i olika riktningar.
- **SCAMPER-metoden:** En systematisk process som innebär att man ställer frågor enligt sju fasta perspektiv: *Substitute, Combine, Adapt, Modify, Put to other uses, Eliminate, Reverse*. Detta möjliggör nya angreppssätt på befintliga idéer eller spelkoncept.
- **Idédagböcker:** Att kontinuerligt dokumentera idéfragment, observationer och associationer kan underlätta i framtida designfaser, särskilt när inspirationen tryter.

Identifiering av marknadsluckor.

Att studera trender och samtida spelutbud kan avslöja områden där efterfrågan är hög men tillgången låg. Det kan röra sig om genrer som förlorat sitt fotfäste eller spelidéer som har potential men ännu inte fått tillräcklig uppmärksamhet. Många framgångsrika indiespel har uppstått genom att fylla just sådana tomrum på marknaden.

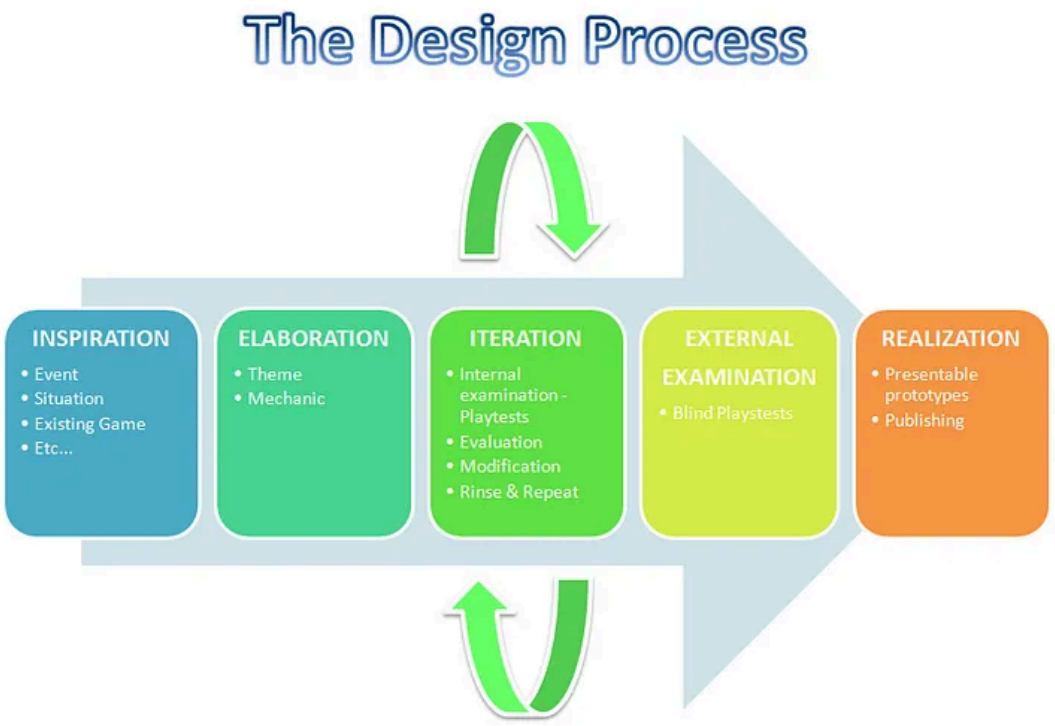
Inspiration från andra medier och verkliga händelser.

Kulturella uttryck såsom film, litteratur, musik och bildkonst har länge inspirerat spelutvecklare. Även historiska händelser, mytologi, samhällsfenomen och personliga erfarenheter kan fungera som värdefulla underlag vid konceptualisering av spelvärldar och berättelser.

Game jams och community-engagemang.

Så kallade *game jams* är tidsbegränsade spelutvecklingstävlingar där deltagarna på kort tid – ofta 48 till 72 timmar – ska skapa ett fungerande spel baserat på ett gemensamt tema. Dessa evenemang uppmuntrar till snabba beslut, improvisation och samarbete, vilket i sin tur ofta leder till oväntade och nyskapande lösningar. Game jams är även ett sätt att snabbt testa och utvärdera idéer i praktiken.

Utöver detta kan aktivt deltagande i spelutvecklingsforum, Discord-servrar och andra digitala gemenskaper bidra med feedback, inspiration och nätverksmöjligheter som är särskilt värdefulla under idéfasen.



Figur 2. Exempel på designprocess för ett spel (Pilakowski, 2017)

Genom att använda dessa metoder och strategier kan man utveckla en spelidé som inte bara är unik utan också har potential att engagera spelare. (Bramble, 2023a)

3.2 Affärsmodell och finansiering

En av de största utmaningarna för indieutvecklare är att finansiera sitt projekt. Utvecklingen av ett spel kan ta månader eller år, och utan en tydlig affärsmodell kan det bli svårt att hålla sig ekonomiskt stabil under processen. Det finns flera sätt att finansiera ett indieprojekt:

3.2.1. Självfinansiering ("Bootstrapping")

Många indieutvecklare finansierar sina spel med egna besparingar, extrajobb eller mindre investeringar. Fördelen är att utvecklaren behåller full kontroll över projektet, men det kan också innebära ekonomisk osäkerhet.

3.2.2. Crowdfunding

Crowdfunding (gräsrotsfinansiering) har blivit ett populärt och effektivt sätt för indieutvecklare att finansiera sina spelprojekt. Det bygger på idén att en stor mängd

privatpersoner – istället för enskilda investerare eller förlag – bidrar med små summor pengar för att stödja ett projekt de tror på. I gengäld får backarna ofta tillgång till exklusivt innehåll, betaversioner eller en kopia av spelet vid lansering.

Plattformar som Kickstarter, Indiegogo och Fig har hjälpt många spelstudior att gå från idé till färdig produkt. Exempelvis finansierades succéspelet Shovel Knight (Yacht Club Games, 2014) via Kickstarter – en plattform som sedan starten 2009 har samlat in över 1,4 miljarder dollar till mer än 65 000 spelprojekt (se figur 3 och figur 4).

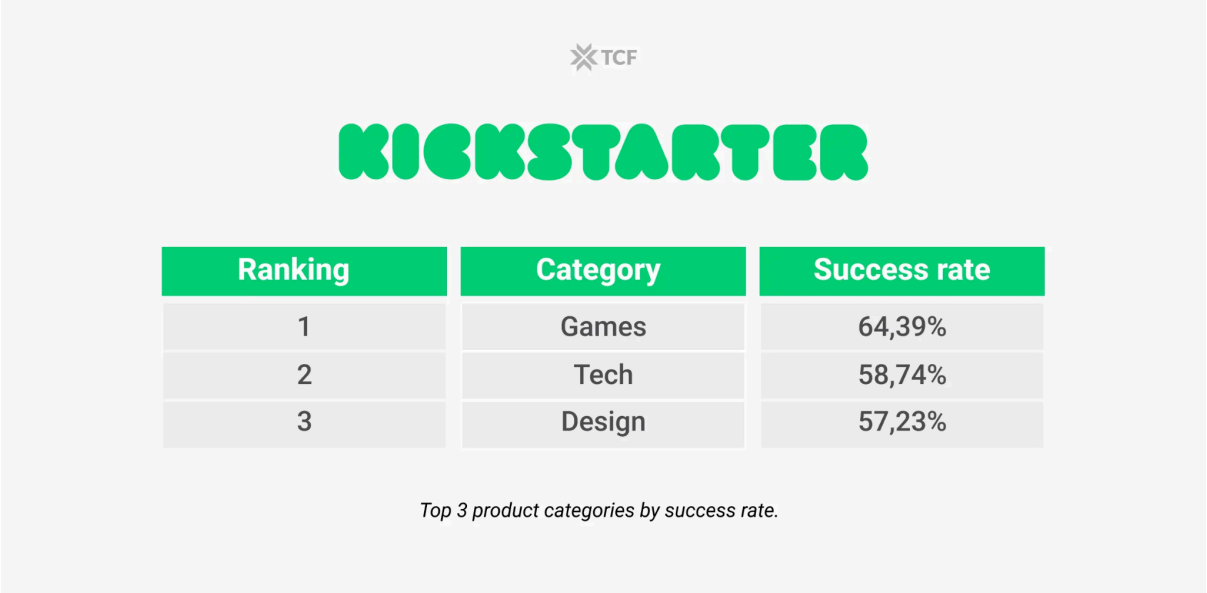
Figur 3. Antal projekt och insamlade pengar per kategori på Kickstarter 2023. "Games" innefattar även brädspel (Zohrabyan, 2023)

Framgångsrika kampanjer bygger inte enbart på en bra spelidé, utan kräver också stark presentation, engagerande trailers och en välplanerad belöningsstruktur. Ett tydligt mål och realistiska tilläggsmål (stretch goals) är avgörande för att vinna spelarnas förtroende, och att hålla en aktiv dialog med sin community under kampanjens gång kan öka chansen för framgång markant.

Statistik visar att Kickstarter-kampanjer som når 20 % av sitt mål inom de första 48 timmarna har en hög sannolikhet att bli fullfinansierade. Samtidigt påminner exemplet med *Mighty No.*

9 (Level-5 Osaka Office, Inti Creates, 2016) om riskerna med att lova för mycket och att inte ha en genomtänkt plan. Spelet, som blev ett av de mest uppmärksammade Kickstarterprojekten med nästan fyra miljoner dollar i insamlade medel, misslyckades med att leva upp till förväntningarna. Efter en stark kampanjstart med legendariske Keiji Inafune vid rodret, väckte projektet enorm entusiasm bland fansen. Men ambitionen att utveckla spelet för ett dussintal plattformar samtidigt och inkludera många tilläggsmål visade sig bli för mycket. Budgeten urholkades snabbt av skatter, plattformsavgifter och kostnader för finansiärbelöningar, vilket i praktiken innebar att långt mindre än fyra miljoner faktiskt gick till utvecklingen. (Ponce, 2016)

Utöver den spridda utvecklingen uppstod kritik mot kommunikationen med finansiärerna, särskilt efter flera förseningar och upplevd brist på transparens. Konflikter kring den anställda community managern och oro för att pengar skulle gå till andra projekt, som den samtidigt annonserade *Red Ash*-kampanjen (också utvecklat av samma studior, men aldrig släppt), ökade misstron. Trots ett starkt team och goda intentioner blev *Mighty No. 9* ett exempel på hur övertro på sina resurser, svårigheter med att hantera communityn och dålig projektledning kan undergräva även de mest lovande crowdfundingkampanjerna. (Ponce, 2016)



Figur 4. De kategorier som oftast nått sitt mål på Kickstarter 2023 (Zohrabyan, 2023)

Sammanfattningsvis är crowdfunding inte bara ett verktyg för finansiering, utan också en möjlighet att testa sin spelidé, bygga ett community och etablera en marknadsföringsplattform inför lanseringen. (Bramble, 2023b)

3.2.3. Early Access

En annan modell är Early Access (tidig åtkomst), där spelet släpps i ofärdigt skick mot betalning. Det ger intäkter under utvecklingens gång och värdefull feedback från tidiga spelare. En empirisk studie av Steam-plattformen 2017 fann att 15 % av spelen på Steam använde Early Access-modellen, och att den absolut populäraste Early Access-titeln hade nått 29 miljoner ägare (Lin et al., 2018). Det spelet tros ha varit *PlayerUnknown's Battlegrounds* (PUBG), som under samma period sålde över 30 miljoner exemplar via Early Access. Fenomenet domineras av mindre studios: 88 % av Early Access-spelen klassades som "indie" av sina utvecklare (Lin et al., 2018). Det antyder att Early Access främst attraherar individer eller små team som behöver tidig finansiering och community-stöd. Early Access har lett till succéer (*Minecraft* och *Kerbal Space Program* började så), men också kritik då över hälften av projekten sägs aldrig nå ett "färdigt" stadium. Förberedelsemässigt måste utvecklare som väljer Early Access planera för kontinuerliga uppdateringar och öppen kommunikation för att behålla spelarnas förtroende under den utdragna utvecklingsfasen.

3.2.4. Förläggare (Publisher)

Många indies söker även partnerskap med förläggare (publishers) eller investerare. Traditionellt sett är det så de flesta större spel blir finansierade; en utvecklare pitchar sin idé till en förläggare som går in med kapital mot rättigheter till en del av intäkterna. Spelveteranen Brenda Romero påpekar att trots indie-boomen är *"det här fortfarande det mest sannolika sättet ett spel blir gjort"* – att man nervöst står i ett mötesrum och försöker övertyga finansiärer att satsa 2–3 år av budget på ens vision (Gordon, 2021). I planeringsskedet innebär detta att förbereda en affärsplan, en prototyp eller demo, och tydligt visa spelets potential (både kreativt och kommersiellt) för att attrahera finansiering. Fördelarna med en publisher kan vara marknadsföringsstöd, distribution, och att

utvecklingsteamet får fokusera mer på själva spelet. Nackdelarna kan vara minskad kreativ kontroll eller krav på vissa designval.

Under 2010-talet har en ny typ av förläggare dykt upp, inriktade på indiespel med hög kvalitet – ibland kallade "Triple-I" (indie-projekt med AAA-ambitioner). Företag som Devolver Digital, Annapurna Interactive och Raw Fury erbjuder finansiering och lansering av indietitlar som vågar vara nischade. Dessa avtal kan se olika ut – ibland finansiering mot en andel av intäkterna, ibland uppköp av IP (immateriella rättigheter), vilket innebär att förlaget eller investeraren köper rätten till spelets koncept, varumärke eller kod. Gemensamt för dessa avtal är att de ger indies access till resurser utan att de själva står för alla kostnader (Gordon, 2021). Utöver detta finns alternativ som statliga kulturstöd eller spelfonder (i vissa länder), samt inkubatorer och acceleratorer som kopplar samman utvecklare med investerare. Sammanfattningsvis måste ett indie-team i planeringsfasen noga utvärdera vilken finansieringsmodell som passar projektet och teamets långsiktiga mål – ofta blir det en kombination (t.ex. Kickstarter för att bygga community + Early Access för fortsatt kassaflöde).

3.3 Team och rollfördelning

Spelutveckling är tvärdisciplinärt och kräver ofta ett team med olika kompetenser. Även om det finns exempel på soloutvecklade spel, är de flesta titlar resultatet av samarbete. En erfaren producent beskrev att det minsta team han arbetat i var tre personer (han själv som designer/projektledare, plus en programmerare och en artist), medan det största var ~40 personer – och i AAA-projekt hos större studior kan hundratals människor vara inblandade (Mullich, 2015). I planeringsstadiet definieras därför vilka roller som behövs för projektet och hur teamet ska organiseras. Ett typiskt utvecklingsteam består av programmerare, grafiker (2D/3D-konstnärer, animatörer), speldesigner samt en producent eller projektledare som samordnar arbetet (Mullich, 2015). Utöver kärnteamet tillkommer ofta specialister: ljuddesigners och kompositörer för musiken, QA-testare (kvalitetssäkringsteam) som kvalitetssäkrar spelet, nivådesigners som bygger banor, och ibland författare eller narrativdesigners för story. I större produktioner finns också tekniska konstnärer, verktygsprogrammerare, community-managers med flera. Varje roll har sina ansvarsområden – t.ex. formulerar speldesignern hur spelet ska fungera i detalj (regler, mål, nivåer, story),

programmerarna implementerar dessa funktioner i kod, och artisterna skapar allt visuellt innehåll (Mullich, 2015). Producentens uppgift är att hålla ihop helheten: planera milstolpar, se till att spelet håller sig inom budget och tidsram, och lösa flaskhalsar så teamet kan arbeta effektivt. En viktig insikt är att även om designern ofta "äger" visionen så är hen inte nödvändigtvis chef – projektledaren balanserar designönskemål mot realiteter (tid, budget) och prioriterar vad som ska göras (Mullich, 2015).

I små indieteam bär en person ofta flera hattar. Det är inte ovanligt att t.ex. en och samma individ är både designer och programmerare, eller att en programmerare också gör ljud och en konstnär sköter marknadsföringen. Denna flexibilitet kan vara nödvändig när resurserna är knappa, men det ställer höga krav på tidsplanering och kompetensbredd. Många indies börjar som en ensam utvecklare och tar sedan in hjälp när projektet växer. Att arbeta ensam vs. i team är ett vägval under planeringen. Soloutveckling har fördelen att man har full kreativ frihet och kan iterera snabbt utan byråkrati – många ikoniska indies (*Minecraft*, *Stardew Valley*, *Undertale*) började som en persons vision. Samtidigt är nackdelarna betydande: allt arbete, från kod till konst till PR(Public Relations/omvärldsrelationer), vilar på en persons axlar, vilket lätt leder till överarbete. Statistik visar också att soloprojekt löper högre risk att aldrig färdigställas jämfört med projekt med team (Wayline, 2025b). Arbete i grupp möjliggör större och mer ambitiösa spel (ingen ensam person hade kunnat göra *Red Dead Redemption 2* (Rockstar Games, 2018)), och man får tillgång till många olika expertis – en bra grupp drar nytta av att varje medlem bidrar med sin specialkunskap (Wayline, 2025b). Dock kräver teamwork tydlig kommunikation, samarbetsförmåga och ibland kompromisser kring den kreativa visionen. I planeringsfasen överväger man således teamets sammansättning noga: vilka roller är kritiska från start, kan vi realistiskt försöka göra detta med två personer eller behöver vi fem, behöver vi outsourca något moment (t.ex. cinematics eller ljud), etc. Att bygga rätt team – eller att inse sina egna begränsningar som soloutvecklare – är avgörande för spelets framdrift.

3.4 Versionshantering och samarbete

Versionshantering är en viktig del av mjukvaruutveckling och underlättar samarbete, kodspårning och hantering av ändringar i projekt. Genom att använda ett

versionshanteringssystem (VCS) kan utvecklingsteam arbeta mer effektivt i distribuerade miljöer och hantera kodändringar smidigt.

Ett VCS förbättrar kodkvaliteten genom att möjliggöra kontinuerlig granskning och samarbete. Genom att spåra varje ändring kan team identifiera och åtgärda problem tidigt, vilket leder till stabilare och mer strukturerad kod.

Utvecklingsprocessen accelereras eftersom VCS tillåter parallellt arbete via grenar och snabb återställning vid problem. Det minskar utvecklings- och felsökningstid, vilket gör att nya funktioner kan levereras snabbare.

Ett centraliserat versionshanteringssystem ökar projektets transparens och gör det enklare att följa förändringar över tid. Teammedlemmar har tillgång till den senaste koden, och integration med projektverktyg kopplar kodändringar till specifika uppgifter och milstolpar. (Gitlab, 2022)

3.4.1 Vad är versionshantering?

Versionshantering spårar alla ändringar i kodbasen, vilket gör det möjligt att återgå till tidigare versioner och förstå kodhistoriken. Detta fungerar som en säkerhetsåtgärd och gör att utvecklare kan experimentera utan risk för permanenta skador. Vid kodkonflikter hjälper versionshantering att identifiera och lösa problem snabbt.

Det finns olika typer av versionshanteringssystem (VCS). Ett centraliserat system (CVCS) lagrar alla filer på en server, medan ett distribuerat system (DVCS), som Git, ger varje utvecklare en kopia av hela historiken. Lock-baserade och optimistiska system används också för att hantera samtidiga ändringar.

Git är den mest populära VCS-lösningen, medan Subversion (SVN) används för centraliserade projekt och Mercurial för enkel skalbarhet. För spelutveckling är versionshantering avgörande för att hantera komplexa projekt, säkerställa samarbete och möjliggöra snabb iteration. (Gitlab, 2022)

3.4.2 Introduktion till Perforce (Helix Core)

Perforce, också känt som Helix Core, är ett centraliserat versionshanteringssystem som är särskilt anpassat för stora projekt med många filer och teammedlemmar. Det används ofta inom spelutveckling på grund av dess effektivitet vid hantering av stora binära filer och dess centraliserade arbetsflöde.

Varför används Perforce i spelutveckling?

1. **Centraliserad versionshantering**
    - Till skillnad från distribuerade system som Git, är Perforce byggt för att hantera stora kodbaser och resursfiler i en enda central server.
    - Detta gör det enklare att hantera rättigheter, synkronisera data och se till att alla teammedlemmar har tillgång till de senaste filerna.
2. **Effektiv hantering av stora filer**
    - Spelprojekt innehåller ofta stora binära filer som texturer, modeller och ljudfiler.
    - Perforce hanterar dessa effektivt genom att endast ladda ned de filer som användaren behöver, vilket snabbar upp arbetsflödet.
3. **Changelists och versionshantering per fil**
    - Varje förändring grupperas i en så kallad "Changelist" som gör det enkelt att se vilka filer som har ändrats och hur de utvecklats över tid.
    - Systemet ger en tydlig överblick över versioner av varje enskild fil, till skillnad från Git som använder commit-hashar för hela projektet.
4. **Streams och grenhantering**
    - Perforce har en avancerad grenhantering genom sitt "Streams"-system som gör det enklare att skapa och underhålla olika versioner av ett spelprojekt.
    - Streams tillåter att förändringar kontrolleras och synkas mellan olika utvecklingsgrenar på ett mer strukturerat sätt än traditionella Git-branches.
5. **Prestanda för stora team och globala projekt**
    - Med stöd för proxy-servrar och edge-servrar kan team som arbetar globalt ha snabba åtkomsttider till filerna.

    - Integreras med Active Directory och LDAP (Lightweight Directory Access Protocol) som är ett standardiserat protokoll som används för att söka och hantera information i ett katalogtjänstsystem.
6. **Integration med spelmotorer och CI/CD (continuous integration/continuous delivery)**
    - Perforce har inbyggda integrationer för spelmotorer som Unreal Engine och Unity.
    - Stödjer kontinuerlig integration och deployment genom integrationer med Jenkins, TeamCity och andra CI/CD-verktyg.

Verktyg i Perforce

- **P4V (Perforce Visual Client)** – En grafisk klient för hantering av filer och changelister.
- **P4Merge** – Ett verktyg för att jämföra och sammanfoga filer.
- **P4Admin** – Ett administrationsverktyg för serverhantering och användaråtkomst.
- **Helix Swarm** – Ett kodgranskningsverktyg för att förbättra kodkvalitet och samarbete.

Nackdelar med Perforce

- **Inlärningskurva**: Det tar tid att lära sig systemet, särskilt för de som är vana vid Git.
- **Komplex serveradministration**: Att installera och konfigurera en Perforce-server kräver teknisk expertis.
- **Branchning och sammanfogning**: Mer komplicerat än i Git, särskilt för flyttade och modifierade filer.

Perforce (Helix Core) är ett kraftfullt versionshanteringssystem som är väl anpassat för spelutveckling tack vare dess prestanda, hantering av stora filer och integrationer med spelmotorer. Trots en viss inlärningskurva och komplex administration, erbjuder det robusta funktioner för storskalig utveckling och samarbete. För team som arbetar med stora projekt och många binära filer kan Perforce vara det bästa valet. (Kahn, 2019)

3.5 Val av spelmotor

Vi valde Unreal Engine eftersom det är en kraftfull motor för 3D-spel, särskilt för högkvalitativ grafik och avancerade system. Det är en av de mest använda motorerna i branschen, vilket innebär att det finns gott om resurser, support och ett stort community.

Alternativ som Unity, Godot och GameMaker Studio har sina egna styrkor, men Unreal erbjuder en kombination av grafik, verktyg och flexibilitet som passar vårt projekt bäst. Unity är ett bra val för både 2D och 3D med en enklare inlärningskurva, Godot är ett lättviktigt och kostnadsfritt alternativ, och GameMaker Studio är optimerat för 2D-spel.

Genom att använda Unreal får vi tillgång till ett omfattande verktyg för att skapa avancerade spelupplevelser med stöd för modern teknik. (Carson, 2024)

4. Praktisk spelutveckling i Unreal Engine

När en spelidé har utvecklats och en plan är på plats börjar den praktiska utvecklingsfasen. I detta kapitel beskrivs arbetsflödet i Unreal Engine, hur man snabbt bygger och itererar prototyper, samt hanterar viktiga tekniska aspekter såsom tillgångar, multiplayer, AI och procedurellt genererade nivåer.

4.1 Grundläggande arbetsflöde i Unreal Engine

Unreal Engine erbjuder två huvudsakliga sätt att programmera spelmekanik: det visuella skriptsystemet *Blueprints* och kod i *C++*. Blueprint är UE:s nodbaserade visuella språk som möjliggör snabb prototypning direkt i editorn, medan C++ ger högre prestanda och mer kontroll över detaljerna (Cowley, 2015). En vanlig arbetsmetod är att designers först bygger prototyper av funktionalitet med Blueprints, varefter programmerare implementerar samma logik i C++ för att få fördelarna av optimerad, native kod (Cowley, 2015). C++-kod kompileras till maskinkod som körs direkt på CPU:n, vilket gör den betydligt snabbare än Blueprint-skript (en studie visade t.ex. att C++ kunde minska energiåtgången per bildruta med upp till ~48 % jämfört med motsvarande Blueprint-logik) (Verón et al., 2024). I praktiken används ofta en kombination – professionella team prototypar och itererar spelfunktioner i Blueprint, för att sedan flytta komplex eller prestandakritisk logik till C++ vid behov (Game Development Stack Exchange, 2024).

4.2 Prototypframställning och iteration

När man prototypframställer spel är det effektivt att börja enkelt. En vanlig metod är *greyboxing*, vilket innebär att man bygger en spelbar grov version av en nivå med enkla geometriska former innan man lägger tid på detaljerad grafik, se figur 5 (Epic Games Developer, n.d.-d). Detta låter utvecklare testa spelmekaniken tidigt och justera designen utan större kostnad. I spelindustrin är det också allmänt accepterat att arbeta iterativt: man skapar en minimal spelbar version (MVP) och förbättrar den i många snabba omgångar. Snabba iterationer och experiment anses vara det bästa sättet att successivt närma sig ett underhållande och välbalanserat spel (Caoili, 2008). Faktum är att spelutvecklare ser iteration som en helt naturlig och nödvändig del av processen – nästan en självklarhet i projektens

tidiga faser (Kultima, 2015). Genom att kontinuerligt bygga ut och justera prototyper (istället för att försöka planera allt i förväg) hittar team “kärnan” i spelet och kan tidigt upptäcka vad som är roligt eller behöver ändras.

Figur 5. Exempel på greyboxing (Mittal, 2024)

4.3 Grafik och tillgångar

Vid utveckling av spelgrafik och 3D-modeller står man inför valet att skapa eget material eller använda färdiga resurser. Att köpa eller ladda ner färdiga tillgångspaket (asset-pack) kan spara mycket tid och pengar, men risken är att spelet tappar i originalitet om många använder samma grafik (Chichmanov, 2024). Egendesignade modeller och texturer ger å andra sidan ett unikt utseende och full kreativ kontroll, men kräver avsevärt mer tid, kompetens och arbete. Indiespelutvecklare måste därför balansera kostnad mot unik identitet – en analys visar att färdiga tillgångar ger snabbare utveckling och lägre kostnad, medan skräddarsytt innehåll gör att spelet sticker ut mer och känns mer autentiskt (Chichmanov, 2024).

När det gäller att använda tredjepartstillgångar är licensiering avgörande. *Creative Commons Zero (CC0)* innebär att verket släpps i public domain – det får användas fritt för alla ändamål, utan att kräva tillskrivning eller tillstånd (OpenGameArt, 2011). Många gratismaterial och open-source-tillgångar använder sådana CC-licenser för att underlätta återanvändning.

Tillgångar från Unreal Marketplace omfattas däremot av Epics egen licens: efter engångsköpet har utvecklaren en livstidslicens att använda innehållet i ett obegränsat antal projekt (utan ytterligare royalties till skaparen) (Epic Developer Community Forums, 2016).

Samtidigt ställer Marketplace-licensen vissa villkor – man får inte återdistribuera eller sälja vidare dessa tillgångar i rå form. Licensen är icke-överförbar och förbjuder t.ex. att köpt innehåll paketeras om och delas med andra utvecklare eller säljs i ett nytt tillgångspaket (Epic Developer Community Forums, 2016). Om man använder tillgångar som är licensierade under GNU GPL (en copyfree-licens främst avsedd för programkod) måste man vara försiktig: GPL ställer krav på att derivatverk också distribueras med öppen källkod. Som påpekats i öppna spelutvecklingsforum är det oklart hur GPL ska tolkas för konst och 3D-modeller, och GPL-licensierade tillgångar kan orsaka konflikter om spelet inte själv är open source (OpenGameArt, 2011). Generellt bör utvecklare säkerställa att tillgångens licens är kompatibel med spelets tänkta distribution och affärsmodell – t.ex. välja royaltyfria (royalty-free) tillgångar för kommersiella projekt för att undvika framtida juridiska problem.

4.4 Multiplayer och nätverkande

Unreal Engine hanterar multiplayer med en klassisk klient-server-arkitektur. Servern, där spelet körs, håller alltid den "sanna", auktoritativa spelvärlden – det är på servern som multiplayerspelet faktiskt utspelar sig. (Epic Games Developer, n.d.-e). Klienterna kopplar upp sig mot servern och får *replikerade* kopior av relevanta objekt (skådespelare/Actors). När servern skapar en aktör som är markerad för replikation, genereras automatiskt motsvarande instans på alla anslutna klienter (Epic Games Developer, n.d.-e). Servern skickar kontinuerligt ut uppdateringar av objektens tillstånd till klienterna, så att alla ser samma spelhändelser synkroniserat. På samma sätt skickar klienterna spelarens input (t.ex. rörelser och handlingar) till servern, där de valideras och påverkar det auktoritativa speltillståndet. Detta server-centrerade upplägg innebär att fusk begränsas (eftersom klienten inte själv kan avgöra spelregler) och att spelare alltid får en konsekvent synkroniserad upplevelse.

4.5 AI i spelutveckling

Artificiell intelligens (AI) är idag en av de mest betydelsefulla teknikerna inom spelutveckling. I spelkontext syftar AI på de system som styr datoriserade karaktärer – så kallade NPC:er (icke-spelbara karaktärer, från engelskans Non-Player Characters)(Wikipedia contributors, 2025b). Dessa system gör att karaktärerna kan navigera, fatta beslut, agera och reagera på spelaren på ett sätt som känns levande, dynamiskt och trovärdigt.

AI kan till exempel få en fiende att jaga spelaren, eller en allierad karaktär att ge hjälp i rätt ögonblick. Men AI i spel skiljer sig från exempelvis AI i självkörande bilar eller konversationsrobotar – här handlar det mindre om att exakt efterlikna mänskligt tänkande och mer om att skapa en engagerande spelupplevelse.

Vad innebär AI i spel? AI i spel är främst designad för att underhålla och utmana spelaren snarare än att uppnå fullständig "intelligens". Den bygger på att skapa illusionen av liv och beslutsfattande genom olika tekniska metoder. Syftet är att förbättra spelupplevelsen genom att få spelvärlden att kännas mer levande, oförutsägbar och reaktiv.

Vanliga användningsområden för AI i spel:

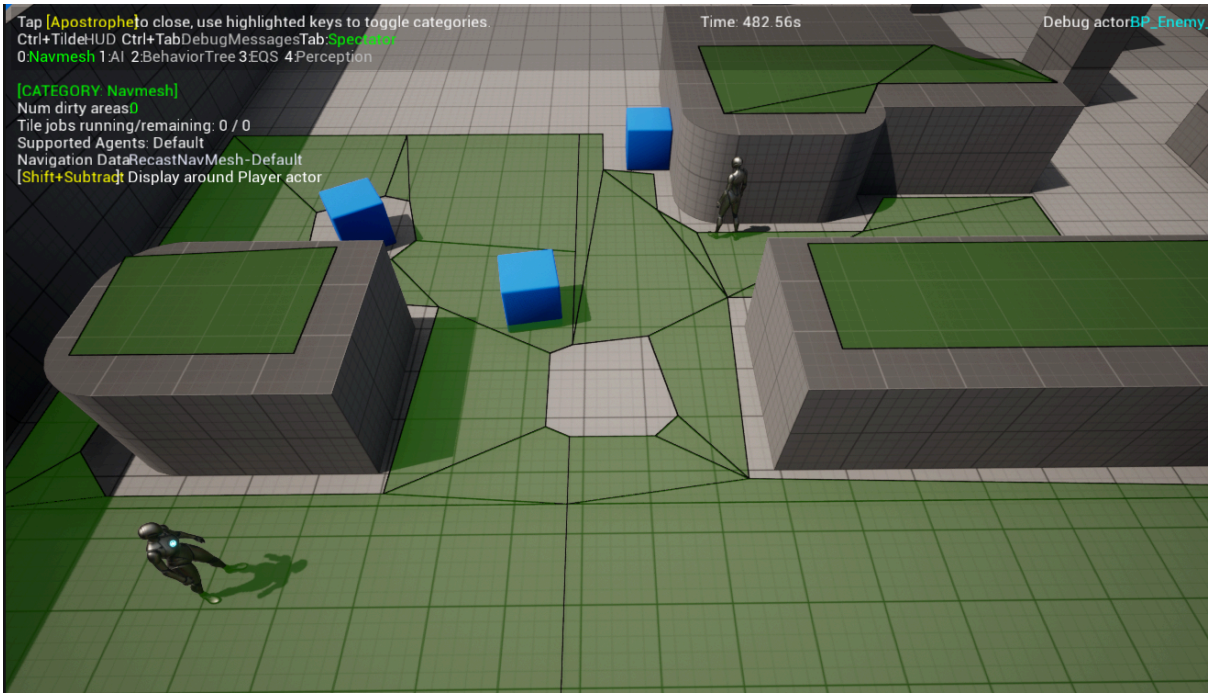
- **Styrning av NPC:er** – AI möjliggör att karaktärer kan gå, prata, slåss, gömma sig eller interagera med miljön.
- **Beslutsfattande** – Karaktärer kan bedöma om de ska anfalla, fly eller gömma sig baserat på situationen.
- **Processuell generering** (Procedural Generation) – AI används för att automatiskt skapa nya miljöer, nivåer eller uppdrag.
- **Dynamisk svårighetsgrad** (Dynamic Difficulty Adjustment) – Spelet kan anpassa sin svårighetsnivå beroende på spelarens prestation. (Nadiy, 2024)

4.5.1 Vägbestämning och navigering

För att en datorstyrd karaktär (NPC) ska kunna ta sig från punkt A till punkt B i en spelvärld, måste den först veta vilka vägar som är tillgängliga och hur man bäst tar sig fram. Detta sker med hjälp av vägbestämning (pathfinding).

En av de mest använda metoderna för detta är A*-algoritmen (A Star) (Belwariar, 2016), som fungerar ungefär som en GPS för spelkaraktärer. Den analyserar alla tänkbara vägar och räknar ut vilken som är kortast eller bäst. Denna algoritm används i nästan alla moderna spel för att få karaktärer att röra sig smart.

För att karaktären ska veta var den kan gå, används ofta något som kallas för NavMesh (Navigeringsnät). Det är som en osynlig karta som spelutvecklaren lägger ovanpå spelvärlden. Den markerar gångbara ytor, till exempel golv och stigar, och ser till att NPC:er inte försöker gå genom väggar eller fastnar i objekt som möbler, se figur 6. (Nadiy, 2024)

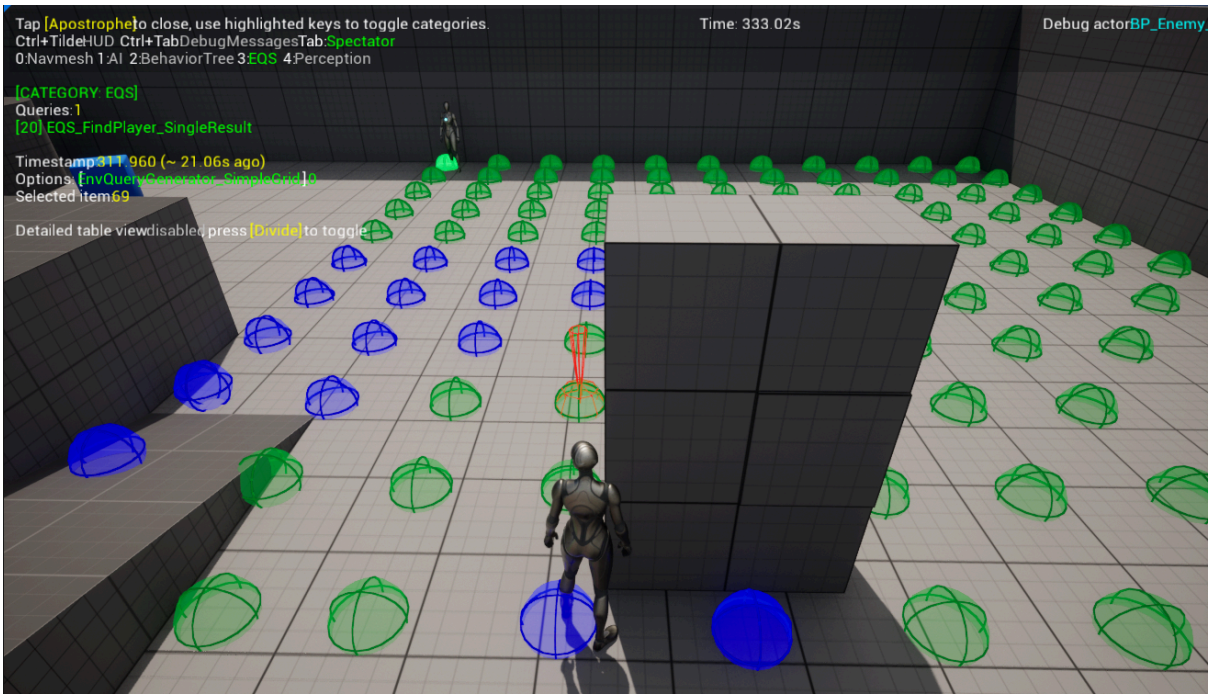


Figur 6. Exempel på NavMesh i Unreal Engine 5 (Epic Games Developer, n.d.-a)

Utöver NavMesh finns även EQS, vilket är ett verktyg som låter AI:n analysera omgivningen för att ta smartare beslut, se figur 7. Den kan ställa frågor som:

- "Var finns bästa skydd från fienden?"
- "Vilken plats ger mig bäst sikt över området?"

Detta är särskilt användbart i taktiska spel där fiender behöver söka skydd eller positionera sig strategiskt i strid. (Epic Games Developer, n.d.-b)



Figur 7. Exempel på EQS i Unreal Engine 5 (Epic Games Developer, n.d.-a)

4.5.2 Beslutsfattande system

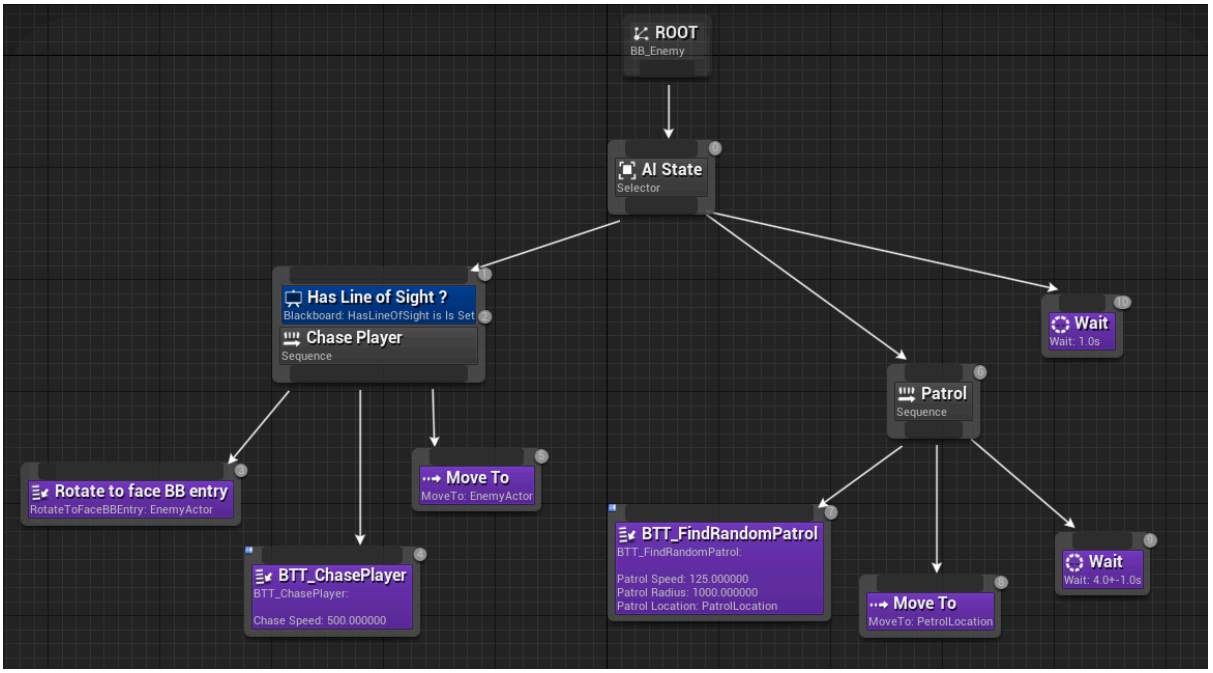
AI i spel måste också kunna fatta beslut – till exempel om en fiende ska anfalla dig, springa därifrån eller gömma sig. Två vanliga tekniker används för att styra detta:

Finita tillståndsmaskiner (FSM – Finite State Machines). En finite state machine bygger på att en NPC alltid befinner sig i ett av flera fördefinierade tillstånd – ungefär som olika "lägen" eller "beteenden". Exempel på sådana tillstånd kan vara:

- Patrullera (röra sig runt i området)
- Jaga spelaren
- Gömma sig
- Gå till anfall

NPC:n växlar mellan dessa tillstånd beroende på händelser i spelvärlden. Om spelaren till exempel upptäcks kan karaktären byta från "patrullera" till "jaga". Det fungerar ungefär som ett trafikljus: varje färg representerar ett visst läge, och systemet bestämmer vilket ljus som ska visas baserat på situationen. På samma sätt styr FSM vilket tillstånd NPC:n ska vara i, beroende på aktuella förutsättningar och stimuli.

Beteendeträd (Behavior Trees). Ett beteendeträd är ett visuellt sätt att styra hur en AI-karaktär ska agera, se figur 8. Det fungerar som ett schema över olika beslut karaktären kan ta, uppdelat i steg och prioriteringar. Exempel: "Om du ser spelaren – jaga honom. Annars – patrullera."



Figur 8. Exempel på beteendeträd i Unreal Engine 5 (Epic Games Developer, n.d.-c)

Till beteendeträdet hör något som kallas för Blackboard – en slags intern minnesanteckning som AI:n använder för att spara viktig information, som "var spelaren sågs senast" eller "hur mycket hälsa karaktären har kvar". Detta gör att karaktären kan fatta beslut baserat på aktuell spelinformation. (Epic Games Developer, n.d.-b)

4.5.3 StateTree

StateTree är ett nyare system som kombinerar två tidigare tekniker: state machines (tillståndsmaskiner) och beteendeträd. Det ger utvecklare ett mer organiserat sätt att bygga AI-logik, särskilt för komplexa karaktärer med många möjliga lägen (t.ex. vila, attackera, springa, gömma sig). Det är också lättare att återanvända och felsöka än ett traditionellt beteendeträd. (Epic Games Developer, n.d.-b)

4.5.4 Adaptiv AI och maskininlärning

I vissa moderna spel försöker AI:n inte bara följa färdiga regler, utan istället lära sig hur du spelar och anpassa sig efter det.

En metod för detta är förstärkningsinlärning (Reinforcement Learning). Det är en form av maskininlärning där AI:n ”testar sig fram” och får poäng (belöningar) för att göra rätt saker. Till exempel kan en AI-styrd motståndare prova olika taktiker och ”lära sig” att du ofta springer åt vänster – och börja förutsäga det.

Det här används till exempel för att justera:

- Hur svår fienden är
- Vilken strategi de använder
- Hur ofta de gör misstag

Syftet är att spelet ska kännas mer dynamiskt och utmanande utan att bli orättvis (Nadiy, 2024).

4.5.5 AI Perception

Det här systemet ger AI-karaktärer "sinnen" – de kan se, höra eller på annat sätt uppfatta vad som händer i deras omgivning. Det går att konfigurera hur långt en NPC kan se, hur känslig den är för ljud, samt vilka stimuli som triggar olika beteenden.

Till exempel kan en vakt missa en smygande spelare som rör sig tyst bakom sig, men om spelaren springer eller avfyrar ett skott kan vakten istället uppfatta ljudet och reagera därefter (Epic Games Developer, n.d.-b).

4.5.6 Smarta Objekt

Detta system gör att AI:n kan interagera med objekt i spelvärlden – till exempel öppna dörrar, sätta sig på en stol eller använda ett fordon. Ett viktigt inslag är reservationssystemet, som förhindrar att flera AI-karaktärer försöker använda samma objekt samtidigt. Det fungerar ungefär som ett kölappssystem för interaktioner. (Epic Games Developer, n.d.-b)

4.5.7 Neural Network Engine (NNE)

Detta är ett avancerat verktyg som låter utvecklare använda neurala nätverk, en form av maskininlärning, direkt i spelet eller i redigeringsverktyget. Med NNE kan AI-karaktärer exempelvis tränas på att känna igen mönster eller fatta beslut utifrån stora mängder data – något som är särskilt användbart i forskning, prototypframställning, eller mycket komplex AI. (Epic Games Developer, n.d.-b)

4.5.8 Processuell generering

AI kan användas för att skapa dynamiska och oförutsägbara spelmiljöer genom processuell generering**.** Detta innebär att nivåer, fiender och utmaningar skapas algoritmiskt snarare än att designas manuellt. Spel som *No Man's Sky* (Hello Games, 2016) och *Minecraft* (Mojang, 2009) använder denna teknik för att skapa unika spelupplevelser varje gång. Det gör att varje spelsession kan kännas unik, och det sparar mycket tid i utvecklingen. (Nadiy, 2024)

Vanliga algoritmer inkluderar t.ex. Perlin noise (Ken Perlins berömda gradientbrusfunktion, flitigt använd för att generera naturtrogna terränger)(Wikipedia contributors, 2025a), dess efterföljare *Simplex noise*, olika *random walk*-metoder (slumpvandring, ibland kallade *"Perlin worms"* när de används för att gräva tunnlar) för att slumpa fram labyrintartade banor, samt cellulära automater som kan generera grottsystem eller utforma kartor av brickbaserade celler (Isaksson, 2024); (Zucconi, 2022). Dessa metoder kan användas var för sig eller i kombination för att bygga upp spelvärldar. När man till exempel använder 3D-Perlinbrus (en sorts slumpliknande mönster) för att skapa grottor, kan man "tröskla"

brusvärdena så att bara de områden där brusvärdet är över eller under ett visst värde blir till tomma utrymmen (grottor), medan resten blir fast mark, eller låta en slumpvandring rita ut sammanlänkade rum i en fängelsehåla. Perlins brusalgoritm har blivit mer eller mindre industristandard sedan 1980-talet – den är mycket vanlig inom all typ av processuell generering och finns inbyggd i många spelmotorer (Zucconi, 2022). Samtidigt utvecklas ständigt nya tekniker (t.ex. L-systems för trädgenerering eller Wave Function Collapse för pusselliknande nivålayout), vilket visar bredden av procedurbaserade metoder inom modern nivå design. Genom att justera parametrar och kombinera algoritmer kan utvecklare uppnå varierade men sammanhängande spelvärldar, och forskningen inom Procedural Content Generation (PCG) erbjuder riktlinjer för vilka metoder som passar olika typer av spelinnehåll (Isaksson, 2024); (Nadiy, 2024).

5. Testning och optimering

Att säkerställa att ett spel är stabilt, responsivt och optimerat är en viktig del av utvecklingsprocessen. Ett spel som innehåller buggar eller prestandaproblem kan leda till en negativ spelupplevelse, vilket i sin tur kan påverka spelets framgång. I detta kapitel behandlas olika testmetoder, verktyg för felsökning samt strategier för att optimera spelets prestanda.

5.1 Buggar och debugging

Buggar – det vill säga fel eller oväntade beteenden i spelet – är en naturlig del av utvecklingsprocessen. De kan visa sig på många sätt, exempelvis genom att figurer fastnar i väggar, menyer inte fungerar som tänkt, eller att två spelare upplever olika saker i ett nätverksspel. Därför är det viktigt att ha metoder och verktyg för att hitta och åtgärda dessa problem. För att hantera buggar effektivt används olika testmetoder och debugging-verktyg.

5.1.1 Testmetoder

Det finns flera olika sätt att identifiera och åtgärda buggar i ett spel:

- **Manuell testning**: Utvecklare eller QA-testare spelar igenom spelet och letar efter fel. Denna typ av testning är särskilt viktig för att upptäcka komplexa fel och för att bedöma spelupplevelsen (Clarke, 2023). QA använder sig ofta av utforskande tester, där testaren fritt navigerar genom spelet för att upptäcka oväntade fel och brister, utan ett strikt manus. De genomför också så kallade "smoke tests", vilket är enkla och snabba tester av grundläggande funktionalitet – till exempel om spelet startar, om menyerna fungerar och om man kan gå in i första nivån – för att tidigt upptäcka allvarliga fel som skulle göra vidare testning meningslös. Nu går det också att testa en stor del med AI vilket kan spara tid och pengar.
- **Automatiserade tester**: Scriptade tester som kontrollerar specifika funktioner, såsom AI beteende eller om spelet korrekt upptäcker när två objekt krockar med varandra. Unreal Engine erbjuder ett Automation Test Framework som stödjer enhetstester,

funktionstester och stresstester. Epic Games automatiserar uppgifter som att ladda alla nivåer för att snabbt upptäcka kraschar.

- **Enhetstester**: Enhetstester är små, fokuserade tester som säkerställer att en specifik del av spelets kod fungerar korrekt, utan att påverkas av andra delar av spelet. Tänk dig det som att kontrollera att en enskild komponent fungerar innan man sätter in det i en maskin. Med C++ i Unreal kan man använda testdriven utveckling (TDD) för att skriva och köra sådana tester.
- **Integrationstester**: Säkerställer att olika system fungerar korrekt tillsammans. I Unreal kan detta inkludera tester som säkerställer att en karta kan laddas och att flera subsystem (t.ex. input, AI och UI) fungerar ihop.
- **Regressionstester**: Efter buggrättningar eller nya funktioner används regressionstester för att försäkra att tidigare fungerande funktionalitet inte har gått sönder. Dessa körs ofta automatiskt i samband med varje ny version.

Debugging-verktyg i Unreal Engine: Debugging, eller felsökning, är processen att identifiera, analysera och åtgärda fel (buggar) i spelets kod eller logik. Det handlar om att förstå varför något inte fungerar som tänkt – till exempel varför en dörr inte öppnas, en fiende inte rör sig, eller att spelet kraschar vid en viss händelse. För att underlätta detta arbete innehåller Unreal Engine flera inbyggda verktyg som hjälper utvecklare att hitta och lösa sådana problem effektivt.

- **Output Log**: En konsollogger som visar felmeddelanden, varningar och utvecklarens egna loggar via UE_LOG eller Print String i Blueprint.
- **Blueprint Debugger**: Möjliggör stegvis felsökning av Blueprintlogik. Genom att sätta brytpunkter pausas exekveringen, och man kan granska variabelvärden och exekveringsvägen.
- **Watch Variables**: Genom att högerklicka på en pin i Blueprint kan man bevaka dess värde under körning. Detta hjälper vid felsökning av dataflöden utan att behöva lägga till loggutskrifter.
- **Stat Commands**: Konsolkommandon som *stat fps, stat unit, stat memory* visar viktig prestandadata. Används för att identifiera flaskhalsar i CPU, GPU m.m. (Corsica, 2020).

- **ProfileGPU (GPU Visualizer)**: Visar hur mycket tid varje renderingspass tar. Verktyget aktiveras med *ProfileGPU*, och ger insyn i t.ex. skuggor, ljus, och post-process-effekters kostnad per bildruta (GPUopen, 2023).

Dessa verktyg och testmetoder kombineras i spelprojekt för att snabbt identifiera och åtgärda fel. De ger inte bara bättre stabilitet utan också förbättrad utvecklingseffektivitet.

5.2 Prestandaoptimering

Optimering är en avgörande del av spelutveckling, särskilt för att säkerställa att spelet körs smidigt på olika system. Dålig prestanda kan leda till låga bildfrekvenser (FPS), långa laddningstider och en allmänt seg spelupplevelse.

5.2.1 Nyckelområden för prestandaoptimering

5.2.1.1 Grafik och rendering

För att spelet ska flyta på bra måste man optimera hur grafiken ritas: en nyckel är att minska onödig geometrisk komplexitet och antalet draw calls – det vill säga de kommandon som skickas från spelet till grafikkortet för att rita varje objekt på skärmen. Level of Detail (LOD) systemet används för att visa förenklade versioner av modeller på långt håll och är särskilt viktigt eftersom små trianglar – det vill säga väldigt små ytor i en 3D-modell som knappt syns på skärmen – påverkar GPU-prestandan oproportionerligt mycket (GPUopen, 2023). Hierarchical LOD (HLOD) är en teknik som kan kombinera flera objekt till en enklare ersättningsmodell, en så kallad *proxy-mesh*, när spelaren är långt bort. Detta förenklar grafiken och gör att spelet bara behöver ge en enda ritinstruktion – en så kallad *draw call* – istället för en per objekt. Genom att minska antalet sådana ritinstruktioner förbättras prestandan markant, särskilt i stora, öppna spelvärldar (GPUopen, 2023).

- **Optimera texturer** – Minska storleken på texturerna (bilder som används för att ge ytor färg och detaljer) och använd *texture streaming*, vilket innebär att spelet bara laddar in högupplösta texturer när de verkligen behövs – till exempel när spelaren är nära ett objekt – för att spara minne och förbättra prestandan.

- **Reducera överflödig skuggning och ljuseffekter** – Dynamiska ljuskällor, som ändrar sig i realtid, kräver mycket beräkningskraft. Därför bör man, där det är möjligt, använda så kallad *baked lighting* – det innebär att ljus och skuggor förberäknas och sparas in i spelets miljöer i förväg, vilket gör att spelet slipper räkna ut ljuseffekterna medan det körs(GPUopen, 2023).
- **Minimera antalet draw calls** – kombinera så kallade *meshes* (3D-modeller som utgör formerna i spelet, t.ex. träd, hus och karaktärer) där det är möjligt, och använd "*instancing"*, vilket innebär att spelet återanvänder samma modell flera gånger utan att behöva skapa en helt ny för varje kopia. Detta sparar både minne och prestanda (GPUopen, 2023).

5.2.1.2 CPU och spelskript

Tick-funktionen i Unreal är ett automatiskt uppdateringsanrop som sker varje bildruta (frame) för varje objekt i spelet – till exempel fiender, dörrar eller ljuskällor. Det innebär att spelet kontinuerligt kör viss kod flera gånger per sekund. Om många objekt använder Tick samtidigt kan det bli mycket krävande för datorn. Därför bör man stänga av Tick där det inte behövs och i stället styra logiken med tidsstyrda funktioner, så kallade *timers* eller *event*, som bara aktiveras vid behov. (Vblanco, 2022).

- **Flytta logik från Blueprint till C++** där möjligt – Blueprint Tick är betydligt långsammare.
- Använd *SetActorTickEnabled(false)* eller sänk Tick Interval för mindre frekventa uppdateringar.
- **Implementera Object Pooling** för att återanvända objekt i stället för att skapa/förstöra dem dynamiskt. Detta minskar minnes fragmentering och CPU-belastning (Larsson, n.d.).

5.2.1.3 Fysik och kollisionshantering

- **Minimera antalet aktiva fysiska objekt** – Använd förenklade krockytor, så kallade *collision primitives*, som till exempel lådor, sfärer eller kapslar, istället för att använda den exakta geometrin från en 3D-modell (så kallad *mesh-kollision*). Primitives kräver

betydligt mindre beräkningskraft och används ofta för objekt där exakthet inte är kritisk, som väggar eller golv.

Se bild nedan, figur 9, för en jämförelse mellan en förenklad primitive och en komplex kollisionsmodell.

Figur 9.Exempel på enkel (vänster) och komplex (höger) krockyta i Unreal Engine 5. Enklare krockytor kan leda till oväntade rörelser, som att karaktären glider (Epic Games Developer, n.d.-f)

- Använd **Fixed Timestep** eller **Async Physics** för mindre kritiska objekt. Profilering med *stat physics* (Epic Games Developer, n.d.-g) och Unreal Insights (Epic Games Developer, n.d.-i) kan avslöja flaskhalsar i fysiksystemet.

5.2.1.4 Laddningstider och minneshantering

- **Level Streaming**: Delar upp världen i mindre nivåer (sub-levels) som laddas vid behov. Unreal 5's World Partition automatiserar detta baserat på spelarens position (Guay, 2024).
- **Asset Preloading** – Ladda in viktiga spelresurser i förväg för att undvika tillfälliga stopp, hack eller fördröjningar under spelandet.
- **Använd instansierade material** – Material som återanvänds minskar minnesanvändningen.

Genom att kombinera dessa tekniker kan utvecklare skapa spel som är mer responsiva, energieffektiva och som fungerar på ett brett spektrum av hårdvara. Optimering bör vara en löpande process under utvecklingen, inte något som sparas till slutet.

5.3 Användartester och feedback

För att säkerställa att spelet är spelbart, engagerande och balanserat är användartester en viktig del av utvecklingen. Spel som testas av verkliga spelare tidigt i utvecklingen har större chans att upptäcka problem som utvecklarna själva kan ha missat.

5.3.1 Typer av playtesting

1. **Intern playtesting** – Utvecklingsteamet testar spelet för att hitta tekniska eller designmässiga problem.
2. **Extern playtesting (stängd beta)** – En mindre grupp spelare utanför teamet får tillgång till spelet och ger feedback – detta ger nya perspektiv och upptäcker problem som teamet själva missat.
3. **Öppen playtesting (Early Access, offentlig beta)** – Spelet släpps till en större publik för att samla in bredare feedback (BackerKit, 2021).

5.3.2 Vad testas under playtesting?

- **Gameplay-balans** – Är spelet för lätt eller för svårt? Finns det frustrerande moment? (Sztajer, 2011)
- **Kontroll och respons** – Känns styrningen naturlig och intuitiv?
- **Buggar och prestanda** – Finns det krascher, prestandadippar eller laddningsproblem?
- **Spelflöde och inlärningskurva** – Är spelet enkelt att förstå, eller behöver tutorials och UI förbättras?

5.3.3 Verktyg för användartester

- **Unreal Insights** – Ett kraftfullt profileringsverktyg som loggar händelser, prestanda, minnesanvändning och mer under tester (Looman, 2022).
- **Heatmaps och spelartelemetri** – Visualiserar var spelare fastnar, dör eller spenderar mest tid. Ubisoft har använt detta för att upptäcka designproblem i nivåer (Dankoff, 2014). Heatmaps visar visuellt *var* något sker i spelet (t.ex. dödsplatser eller rörelsemönster), medan spelartelemetri är den underliggande datainsamlingen som även kan innehålla exakta siffror, tider och val spelare gör.
- **Feedback-formulär och Discord-server** – Möjliggör direkt kommunikation med spelarna.

Playtesting är en iterativ process där varje testomgång används för att göra förbättringar. Interna tester kan ske dagligen inom teamet, medan större externa tester sker vid viktiga milstolpar. Kombinationen av mänsklig feedback och datadriven analys ger utvecklaren en heltäckande bild av spelets tillstånd. Som branschforskning påpekar: spelutvecklare har idag möjligheten att skapa bättre spel inte bara genom magkänsla, utan genom faktiska mätdata om hur spelet upplevs i praktiken (Powell, 2016).

6. Marknadsföring och lansering

6.1 Marknadsföringsstrategier för indiespel

Att nå ut i bruset är en stor utmaning för indieutvecklare. Nedan följer några beprövade marknadsföringsstrategier och verktyg, med fokus på kostnadseffektiva metoder relevanta för digital distribution.

6.1.1 Utvecklingsbloggar (devlogs)

Att föra en utvecklingsblogg (devlog) kan vara ett kraftfullt verktyg för indieutvecklare. Genom regelbundna inlägg om spelutvecklingen kan utvecklare bygga upp ett community av intresserade följare långt före spelets lansering. En studie om digital marknadsföring för indiespel noterar att de flesta indieutvecklare använder utvecklingsbloggar, forumtrådar eller till och med livestreaming av utvecklingen för att *"få social uppmärksamhet"* kring sitt projekt (Wu, 2017). Denna kontinuerliga insyn i utvecklingen skapar engagemang, fans känner sig delaktiga och investerade i spelets resa, vilket kan öka det långsiktiga intresset.

Utvecklingsbloggar fungerar också som historik över projektets framsteg och kan ge upphov till "hype" genom att visa upp nya funktioner, konst eller prototyper. Genom att dela både framgångar och motgångar bygger utvecklaren förtroende och äkthet i spelkulturen. Allt eftersom tiden går kan en lojal fanskara formas, vilka inte bara sprider ordet om spelet utan också står redo att stötta vid lansering. Kort sagt, utvecklingsbloggar hjälper indiespel att *bygga upp följare* genom transparens och regelbunden kommunikation (Wu, 2017). Detta förberedande community kan vara avgörande vid lansering, en engagerad fanskara innebär fler omedelbara spelare, recensioner och rekommendationer till andra.

6.1.2 Trailers och visuellt material

Visuella tillgångar som trailers, skärmdumpar, GIF:ar och konceptbilder – är bland de viktigaste marknadsföringsverktygen för indiespel. Flera marknadsföringsguider betonar att *"trailern är din bästa tillgång för att väcka intresse"*, då den kombinerar snygga bilder, animation, ljud och musik för att snabbt förmedla spelets känsla (Lovato, 2017). En välgjord trailer kan på två minuter ge publiken en stark första upplevelse av spelets atmosfär och

spelupplevelse. Forskning visar också att visuellt innehåll engagerar publik mer effektivt än text, eftersom människor bearbetar visuella intryck mycket snabbare och ofta minns dem bättre (Plugova, 2021).

För att en trailer ska vara effektiv finns det några prövade principer att följa (Lovato, 2017):

- Håll den kort – runt 60–90 sekunder är optimalt. Längre trailers riskerar tappa tittarens uppmärksamhet (Lovato, 2017).
- Fånga intresset direkt – de första fem sekunderna bör innehålla något slagkraftigt som krokar tittaren (Lovato, 2017) (t.ex. ett imponerande visuellt inslag eller en actionfylld scen). På internet scrollar publiken snabbt vidare om inget fångar dem omedelbart.
- Fokus på gameplay – visa genuina spelsekvenser tidigt. Publiken "vill se ditt spel" och förstå hur det spelas (Lovato, 2017). Undvik att gömma spelet bakom långa logotyper eller endast spelfilmsekvenser.
- Starka ljud och musik – ett bra soundtrack och ljudeffekter förstärker trailerns emotionella genomslag (Lovato, 2017). Ljudbilden ska spegla spelets stämning och höja helhetsintrycket.
- "Call to action" i slutet – avsluta trailern med tydlig information om spelets release, plattform eller önskelista (Lovato, 2017). Tala om för tittaren vad nästa steg är (exempelvis "Tillgängligt på Steam eller lägg till i önskelista nu!").

Utöver trailers bör indieutvecklare kontinuerligt dela annat visuellt material under utvecklingens gång. En kvalitativ studie av indiestudios visade att framgångsrika team ofta publicerar skisser, konceptkonst, skärmbilder och GIF:ar under utvecklingens gång, vilket *gradvis attraherar allt fler intresserade* (Plugova, 2021). Till exempel beskrev ett utvecklingsteam sin visuella marknadsföring som "vårt starkaste vapen för att bli upptäckta" (Plugova, 2021). De använde en distinkt konststil och delade kontinuerligt konst och gameplay-bilder på Twitter och Instagram. Detta drog successivt till sig fler följare och potentiella spelare. Sammantaget gäller att en konsekvent visuell närvaro – från en proffsig trailer till regelbundna bilduppdateringar – hjälper spelet sticka ut i bruset och håller publiken nyfiken över tid.

6.1.3 Plattformsspecifik synlighet (Steam och Epic Games Store)

Att förstå hur man utnyttjar olika plattformars system är avgörande för att öka synlighet och försäljning. Steam dominerar PC marknaden och har sina egna algoritmer och funktioner som indieutvecklare bör känna till. En viktig faktor på Steam är *önskelistor*. Många utvecklare använder lanseringsönskelistor som en indikator på förväntad försäljning och med god anledning. Data tyder på att det under första veckan efter release konverteras ungefär 10–20 % av de som önskat spelet till faktiska köpare (Denby, 2022). Med andra ord, om 10 000 personer har satt upp spelet på sin Steam-önskelista vid lansering, kan man grovt estimera 1 000–2 000 sålda exemplar första veckan (Denby, 2022). Dessutom har spelutvecklaren och plattformsägaren **Valve** – företaget bakom Steam – bekräftat att önskelistor påverkar Steams upptäcktsalgoritm innan lansering. Många önskningar och mycket trafik till produktsidan gör att spelet oftare dyker upp för andra användare på plattformen (Denby, 2022). Att aktivt driva trafik till Steam sidan och uppmana till "Lägg till i önskelista" i marknadsföringen är därför en etablerad strategi bland indieutvecklare.

Steam belönar också utvecklare som håller sin community engagerad via uppdateringar och evenemang. Valve erbjuder t.ex. *Update Visibility Rounds*, vilket innebär att när en utvecklare släpper en större uppdatering eller nyhet om spelet, får spelet extra exponering på Steams förstasida för alla som äger eller har önskelistan spelet (Steamworks, n.d.-d). Detta är ett sätt att påminna följare och intresserade om spelet och kan leda till en försäljningsboost vid uppdateringar. Vidare arrangerar Steam regelbundet festivaler (som *Steam Next Fest*) där hundratals demoer av kommande spel lyfts fram under en begränsad tid. Dessa evenemang har visat sig ge betydande genomslag för indietitlar. Medianresultatet för deltagande spel är ungefär en fördubbling av deras önskelistor under denna vecka (Sharman, 2024). Många spel får mellan 1 000 och 3 500 nya önskelistor tack vare exponeringen under ett enda Next Fest (Sharman, 2024). Det råder därför enighet i branschen om att *"festivaler är fantastiska"* möjligheter för indiespel att synas (Zukalous, 2021). Deltagande i sådana evenemang, samt andra Steam-kampanjer (reaktioner, reor, tidsbegränsade demoer), bör alltså vara en del av indieutvecklarens lanseringsstrategi.

Epic Games Store (EGS) skiljer sig från Steam i både publik och funktioner, men erbjuder egna chanser till synlighet. Epic har färre sociala funktioner och algoritmiska flöden, så

synligheten beror ofta på *kuratering*. Ett sätt att snabbt nå ut på EGS är genom Epics initiativ att erbjuda exklusivitetsavtal eller gratisveckor. Som PC Gamer noterade är *"den gyllene biljetten"* för indies på Epic att antingen sluta ett exklusivitetsavtal eller bli utvald som veckans gratis spel (Zak, 2020). Exklusivitet innebär att spelet lanseras på EGS före andra plattformar, i utbyte mot ekonomisk garant (så att utvecklaren täcker sina kostnader) och ofta framträdande placering på butiken. Utvecklaren Roberto Semprebene menar att Epic-exklusiviteten för deras spel *Close to the Sun* (Storm in a Teacup, 2019) garanterade att studion gick runt ekonomiskt, även om den totala försäljningen blev lägre än om de hade släppt på Steam direkt (Zak, 2020). Epic ger alltså *förskottsintäkter och trygghet* till indieutvecklare i sådana avtal.

Att bli veckans gratisspel på EGS kan dessutom ge enorm exponering. När *Close to the Sun* senare gjordes gratis under en vecka på Epic Games Store fick spelet *"otrolig skjuts"* i uppmärksamhet plötsligt ett år efter release, skrev mängder av media, subreddit-trådar och spelsajter om spelet igen eftersom det var på EGS förstasida som gratiserbjudande (Zak, 2020). Denna typ av publicitet är svår att uppnå på egen hand som liten studio. Även om man förstås ger bort spelet gratis under kampanjen, kan effekten bli många nya spelare (miljontals nedladdningar är inte ovanligt för Epics gratis titlar) samt ökad försäljning av DLC (Downloadable Content) eller uppföljare. Sammanfattningsvis bör indieutvecklare som lanserar på PC överväga en plattformsspecifik strategi: använda Steams verktyg, såsom önskelistor, uppdateringar och festivaler, för att maximera algoritmisk synlighet och community hype, samt utnyttja Epic Games Stores möjligheter genom eventuella avtal eller kampanjer som kan ge bred exponering (Zak, 2020).

6.1.4 Sociala medier och PR (Public Relations)

Sociala medieplattformar och PR aktiviteter är nyckeln till att driva extern trafik och bygga upp ett fan community utanför själva butiksplattformarna. Indieutvecklare använder flitigt Twitter (numera X), Reddit, Discord samt samarbeten med influencers/bloggar för att marknadsföra sina spel. Varje kanal har sin roll: Twitter/X är en öppen arena för att nå ut brett med korta uppdateringar, Reddit erbjuder nischade communityn med genuint intresse,

Discord skapar en närkontakt med de mest engagerade fansen, och mindre spelinriktade bloggar eller YouTube-kanaler kan ge personlig och riktad exponering.

Studier av indiemarknadsföring visar att framgångsrika utvecklare ofta uppdaterar sociala medier dagligen under utvecklingen med visuellt innehåll och dialog. En indieutvecklare vittnade till exempel om att dagliga Twitter-inlägg med nya bilder, samt aktiv närvaro på Instagram, betydde mycket för att hålla publiken engagerad och få feedback (Plugova, 2021). Samma utvecklare drev också en Discord-server för de mest hängivna fansen, och använde Reddit som *"en väldigt pålitlig plattform för att bygga en fanskara och bli uppmärksammad"* (Plugova, 2021). Den kombinerade strategin att sprida innehåll brett på öppna sociala medier och samtidigt odla en kärncommunity på Discord ger både räckvidd och djupare engagemang. Reddit, med forum som r/gamedev eller subreddits för specifika genrer, används ofta för att dela utvecklingsframsteg, alpha-/beta-versioner eller AMA (Ask Me Anything) trådar, vilket kan locka intresse från just den målgrupp spelet siktar på.

Att involvera influencers och speljournalister i marknadsföringen kan verka skrämmande för en indie, men fokus ligger ofta på de mindre, nischade aktörerna snarare än bara de största namnen. Mindre YouTube-kanaler eller Twitch-streamers som är inriktade på indies eller på en viss genre kan vara mer benägna att ge okända spel en chans. En rekommenderad taktik är att skicka personliga pressmeddelanden och nycklar (spellicenser) till sådana skapare och till spelbloggar för att få täckning (Lovato, 2017). Traditionell press, det vill säga de stora spelsajterna, kan vara svår att bryta igenom för ett litet spel, men nischbloggar och mikroinfluencers har ofta en hängiven följarskara som litar på deras omdömen. Genom att tidigt bygga relationer, till exempel genom att skapa en särskild Discord för press och streamers eller erbjuda exklusivt material, kan utvecklare få värdefull exponering inför en relevant publik utan stor budget.

Det finns dock lärdomar av tidigare indiekampanjer som indikerar att inte alla sociala kanaler ger lika stor utdelning. En genomgång av "indie post-mortems" (reflektioner över tidigare spelutvecklingsprojekt) 2021 visade att många utvecklare ansåg effekten av Twitter marknadsföring och pressreleaser vara ganska begränsad, men till exempel direkt community engagemang och streamers gav större utslag (Zukalous, 2021). Samtidigt fann en akademisk studie (Golding et al., 2022a) att de indiespel som lyckades passera Steams osynlighetströskel

(få över 10 recensioner och därmed trigga algoritmens exponering) nästan alltid hade använt Twitter aktivt i sin marknadsföring, till skillnad från många av de som förblev obemärkta (Golding et al., 2022b). Den mest troliga tolkningen är att sociala medier måste användas strategiskt, det räcker inte att tweeta ibland, utan det krävs en genomtänkt kampanj med rätt hashtags, regelbundenhet och interaktion med andra. Kampanjer som #WishlistWednesday på Twitter, utvecklardagböcker på YouTube, eller att dela framsteg på populära forum som TIGSource eller Itch.io:s community kan alla bidra till att bygga uppmärksamhet kring spelet.

PR för indiespel handlar ofta om att göra sig hörd utan att ha en stor budget. Det mest effektiva sättet är därför att nå ut genom engagemang i sociala medier och spelcommunityn utan att betala för annonser. Lyckade indieutvecklare kombinerar ofta flera kanaler, de engagerar sin community direkt (via Discord-chattar, utvecklingsuppdateringar på Steam/itch.io), de postar engagerande innehåll på Twitter och Reddit för att dra in nya följare, och de nätverkar med andra utvecklare och journalister (t.ex. genom gemensamma Twitter Spaces, dev streams eller gästinlägg på varandras bloggar). Genom att vara närvarande och genuin på dessa plattformar bygger utvecklaren ett varumärke kring sig själv och spelet. Så när lanseringsdagen närmar sig har många redan hört talas om spelet flera gånger i sina flöden. I slutändan är nyckeln att använda sociala medier och PR till att få folk att bry sig om spelet innan det släpps, då ökar chansen dramatiskt att de faktiskt köper och spelar det vid release (Denby, 2022); (Wu, 2017).

6.2 Steam och Epic Games Store

Steam och Epic Games Store är två av de ledande digitala distributionsplattformarna för PC spel. Båda erbjuder självutgivning för indieutvecklare, men genom olika system (Steamworks respektive Epic Games Developer Portal) med sina egna krav, kostnader och verktyg. Nedan följer en akademisk översikt av hur lanseringsprocessen fungerar på vardera plattform, med fokus på avgifter, lanseringsverktyg, synlighetsmöjligheter och utvecklingskit (SDK).

6.2.1 Steamworks

Anmälan och avgift: För att lansera ett spel på Steam måste utvecklaren gå med i Steamworks (Valves partnerprogram för distribution). Detta innefattar att fylla i digitalt pappersarbete (inklusive skatte- och bankinformation för utbetalningar) samt betala en Steam

Direct avgift på cirka 100 USD per spel. Denna avgift tas ut för varje enskild titel som ska distribueras och fungerar som en tröskel för att motverka spam och lågkvalitativa spel. Avgiften är icke-återbetalningsbar, men Steam låter utvecklare få tillbaka beloppet (recoup) efter att spelet tjänat minst 1 000 USD i intäkter. När avgiften är betald och nödvändiga avtal (t.ex. Steam Distribution Agreement) accepterade, får utvecklaren tillgång till Steamworks gränssnittet för att hantera spelets sida. (Xsolla, 2021b)

Förberedelse av butikssida och integrationer: Via Steamworks kan utvecklaren skapa och konfigurera spelets butikssida (med beskrivningar, media, pris osv.), ladda upp byggen samt ställa in eventuella Steam-funktioner innan lansering (H. Games, 2023). Valve kräver att nya titlar har en publikt synlig "Coming Soon" sida i minst två veckor före lansering, så att spelare kan önskelista spelet och bygga förhandsintresse (H. Games, 2023). Under förberedelsen kan utvecklaren också integrera Steam specifika funktioner genom Steamworks SDK (Software Development Kit). Steamworks SDK erbjuder ett brett utbud av API:er (Applikationsprogrammeringsgränssnitt: Används för kommunikation mellan olika mjukvaror) för att förbättra spelupplevelsen, till exempel kan spelet utnyttja Steam Cloud för att spara spelfiler på Steams servrar (så att användare kan komma åt sina sparfiler från olika datorer) (Steamworks, n.d.-c). Vidare kan Steam Stats and Achievements API användas för att införa prestationer (achievements) och statistik som kopplas till spelarens Steam profil (Steamworks, n.d.-b). Andra funktioner som stöds inkluderar matchningssystem för multiplayer, Steam Workshop (för användarskapat innehåll) och mycket mer, vilka alla är tillgängliga via Steamworks SDK.

Granskning och lansering: Innan spelet kan släppas genomgår det en kortfattad granskningsprocess hos Valve. När utvecklaren anser att både spelklienten och butikssidan är redo för publikation, skickas allt in för granskning via Steamworks. Valves team utför då en grundläggande kontroll: de startar spelet, ser över att butikssidan är korrekt ifylld, och verifierar att spelet kan installeras och köras utan problem för användarna (H. Games, 2023). Denna granskningsprocess tar vanligtvis omkring 1–5 arbetsdagar (H. Games, 2023). Utöver granskningen tillämpar Steam tidskrav för nya releaser, det måste gå minst 30 dagar från det att utvecklaren betalat registreringsavgiften tills spelet faktiskt får lanseras på plattformen (H. Games, 2023). Detta ger Valve tid att validera utvecklarens information och ger utvecklaren

tid att förbereda lanseringen ordentligt. När både tidsfristen och granskningen är avklarade kan utvecklaren själv schemalägga och genomföra spelets lansering, Steam ger kontrollen över lanseringsdagen till utvecklaren när alla villkor är uppfyllda. På lanseringsdagen distribueras spelet genom Steam klienten till alla köpare, och spelets butikssida öppnas för försäljning globalt enligt utvecklarens pris- och regioninställningar.

Intäktsfördelning: Efter lansering sköter Valve alla transaktioner och distribution. Standardmodellen för intäktsfördelning på Steam är 70/30, där utvecklaren får 70 % av intäkterna och Valve behåller 30 % som provision (Xsolla, 2021b). Denna fördelning är en branschstandard för digitala marknadsplatser. För mycket framgångsrika titlar har Valve infört trösklar som kan öka utvecklarens andel, till exempel vid över 10 miljoner USD i intäkter sjunker Valves andel, men för de flesta indiespel är 70/30 modellen som gäller. Intäkterna utbetalas regelbundet till utvecklarens angivna bankkonto, efter avdrag för skatter, moms och eventuell återbetalningshantering enligt Steams avtal.

Synlighetsmöjligheter på Steam: En viktig del av lanseringssprocessen är hur spelet exponeras för Steam-användare. Varje nytt spel på Steam får automatiskt en initial lanseringssynlighet. Valve tillämpar det som internt kallas en "launch visibility round", vilket innebär att det nysläppta spelet under en period lyfts fram på olika delar av butiken (Handrahan, 2020). Spelet dyker upp i listor som *Nya Släpp* och *Nytt & Populärt* (för sin genre och generellt), och kan vid tillräckligt intresse även synas på Steams förstasida för vissa användare (Handrahan, 2020). Dessutom börjar spelet inkluderas i användares personliga discovery queues, vilket ger ytterligare exponering, varje Steam-användare får dagligen se ett urval av nya titlar i sin “upptäckskö” (Discovery Queue), anpassat efter deras preferenser, där varje spel normalt visas endast en gång per användare (Handrahan, 2020).

6.2.2 Epic Games Store Developer Portal

Översikt och registrering: Epic Games Store (EGS) introducerades 2018 som en konkurrent till Steam, och har sedan dess lockat miljontals användare. Till en början var EGS kuraterad, endast utvalda partners fick lansera spel men i mars 2023 öppnade Epic plattformen för självutgivning av spel. Epic lanserade då nya lanseringsverktyg i sin Epic Developer Portal som gör det möjligt för *alla* utvecklare och utgivare att själva släppa spel på EGS, förutsatt

att spelet uppfyller vissa kvalitetskrav (E. Games, 2023). För att börja lansera måste utvecklaren skapa ett konto på Epic Developer Portal och genomgå registreringsproceduren (ålderskrav 18+, juridiska avtal etc., liknande processen på Steam). En registreringsavgift tas ut även här, Epic kräver en standardavgift på 100 USD per spel som ska lanseras (Obedkov, 2023). Avgiften är densamma som Steams och syftar till att täcka administrationskostnader samt avskräcka opålitliga projekt. Till skillnad från Steam nämner inte Epic något om att avgiften kan återfås vid en viss intäkt, den får betraktas som en ren registreringskostnad. När kontot är satt upp och avgiften betald får utvecklaren tillgång till Epic Games Developer Portal, ett webbgränssnitt för att hantera spelets lansering.

Användning av Developer Portal**:** I Epic portalen kan utvecklaren ställa in allt som krävs för lansering av spelet. Detta inkluderar att bygga en butikssida för spelet (komplett med beskrivning, bilder/video, systemkrav, pris och liknande) samt eventuellt sätta upp en "Coming Soon"**-**sida för att samla intresse innan release (Obedkov, 2023). Utvecklaren laddar även upp spelets byggen/installationer via portalen, det finns stöd för att först ladda upp testbyggen för kvalitetssäkring. När all information är ifylld och ett bygge uppladdat kan projektet skickas in för granskning hos Epic (Obedkov, 2023). Epic utför en innehålls- och kvalitetskontroll av spelet och dess butikssida innan godkännande, i syfte att säkerställa att samtliga krav efterlevs. Hela processen sker i självbetjäningsform: *”[Utvecklarna] behöver inte arbeta direkt med oss; de kan själva uppdatera sin närvaro och innehåll,”* förklarade Epic Games VD Tim Sweeney i samband med lanseringen av portalen (Obedkov, 2023). När spelet klarat granskningen får utvecklaren möjlighet att sätta ett lanseringsdatum och lansera spelet på EGS, där det blir tillgängligt för butikens användare.

Krav och kvalitetsstandarder: Till skillnad från Steam – som har relativt få förhandskrav utöver legala och tekniska formaliteter, ställer EGS upp tydliga lanseringskrav som varje spel måste uppfylla för att accepteras på plattformen. De viktigaste kraven är (Obedkov, 2023):

- Crossplay för PC: Om spelet innehåller online multiplayer kräver Epic att PC crossplay stöds mellan samtliga PC butiker (E. Games, 2023). Det innebär att en spelare som köpt spelet på EGS ska kunna spela online med en vän som köpt spelet via t.ex. Steam, och vice versa. Syftet är att inte låsa in spelarbasen till en viss butik. Epic erbjuder utvecklare gratis verktyg för att uppnå detta, t.ex. genom Epic Online

Services, men det står utvecklaren fritt att implementera crossplay på valfritt sätt så länge kravet uppfylls (E. Games, 2023).

- Achievements**:** Spel som onboardas (läggs till) via Developer Portal efter mars 2023 måste aktivera Epic Games Store Achievements *om* spelet har prestationer på någon annan PC plattform (E. Games, 2023). Detta krav standardiserar spelarnas upplevelse, om ett spel på Steam har prestationer för olika milstolpar ska motsvarande finnas för Epic-spelarna. Epic tillhandahåller ett prestationssystem i sin SDK som utvecklaren kan integrera.

- Åldersgräns (IARC): Epic kräver att utvecklaren anger åldersrekommendationer för spelet enligt officiella åldersgränser. För att underlätta har Epic integrerat IARC (International Age Rating Coalition) direkt i portalen, så att utvecklare enkelt kan skaffa åldersmärkningar (ESRB, PEGI, med flera) kostnadsfritt via ett formulär (E. Games, 2023). Om spelet redan har en officiell åldersgräns (till exempel om det släppts på konsol) måste denna visas på spelets EGS sida (E. Games, 2023).

- Förbjudet innehåll: Vissa typer av innehåll är *inte tillåtna* på Epic Games Store. Epics policy förbjuder bland annat hets mot folkgrupp eller annan hatfylld/diskriminerande text, pornografiskt innehåll, olagligt innehåll samt material som inkräktar på andras immateriella rättigheter (E. Games, 2023). Även bedrägerier, “scams” eller skadlig kod förkastas. Denna riktlinje liknar Steams grundläggande regler, Steam tillåter visserligen vuxet innehåll under strikta filter, men förbjuder till exempel olagligheter och uppenbart stötande material (Xsolla, 2021b). Epic utövar kvalitetskontroll baserat på dessa policyer och kan neka spel som bryter mot dem.

- Grundläggande kvalitet: Spel som lanseras måste hålla en rimlig kvalitetsnivå. I praktiken innebär det att spelet ska starta, installera och köras på ett sätt som överensstämmer med vad som utlovas på butikssidan (E. Games, 2023). Om ett spel exempelvis är extremt buggigt eller inte motsvarar beskrivningen, förbehåller sig Epic rätten att avslå det under granskningsprocessen (E. Games, 2023). Målet är att upprätthålla en bra användarupplevelse för EGS-kunder. Skulle ett inskickat spel inte

möta kraven ovan, initierar Epic en utredning och meddelar utvecklaren; spelet måste då åtgärdas eller riskerar att inte få lanseras (Obedkov, 2023).

Intäktsfördelning och ekonomiska villkor: En av de mest omtalade aspekterna med Epic Games Store är dess gynnsamma intäktsfördelning för utvecklare. EGS tillämpar en 88/12-modell, vilket innebär att utvecklaren behåller 88 % av intäkterna medan Epic bara tar 12 % i kommission (E. Games, 2023). Detta kan jämföras med Steams 70/30-modell – skillnaden innebär att av varje såld kopia får utvecklaren behålla en större andel på EGS än på Steam. För spel som dessutom är gjorda i Unreal Engine finns ytterligare en ekonomisk fördel: Epic avstår från spelmotorroyalty på försäljning via Epic Games Store (E. Games, 2023). Normalt tar Epic Games 5 % royalty på intäkter över 1 miljon USD för spel som använder deras Unreal-motor, men försäljning som sker genom EGS med Epics egen betalningslösning undantas från denna avgift (E. Games, 2023). Det betyder i praktiken att en Unreal-baserad titel som säljs på EGS endast betalar den 12-procentiga butiksavgiften och inga extra spelmotoravgifter, vilket är ekonomiskt fördelaktigt för utvecklaren. Precis som på Steam sker utbetalningar av intäkter regelbundet, och Epic hanterar globala skatter, återbetalningar m.m. enligt sitt avtal med utvecklaren.

Plattformsfunktioner och tjänster: Epic Games Store erbjuder ett växande utbud av plattformsfunktioner för utvecklare, men inte än lika omfattande som Steams ekosystem. Grundläggande funktioner som vänlistor, matchmaking och sociala system stöds via Epics infrastruktur, och molnsparningar (cloud saves) finns implementerade för många spel på EGS (Epic aktiverar detta i samarbete med utvecklaren per spel). Prestationssystemet på EGS är, som nämnt, numera tillgängligt och kan integreras för att spegla prestationer i spelet. Epic erbjuder även en del unika verktyg, till exempel ett ”Support-A-Creator”-program där utvecklare kan koppla influencers/streamers som anslutna marknadsförare och dela en liten andel av intäkten med dem som belöning för att de lyfter fram spelet (E. Games, 2023). Detta kan öka exponeringen utanför själva butiken genom att motivera innehållsskapare att marknadsföra spelet. En annan fördel Epic lyfter fram är att de kan hjälpa till med professionell lokalisering av butikssidor till flera språk utan extra kostnad (utvecklaren kan begära detta via supportärenden i portalen) (E. Games, 2023), vilket underlättar för små team att presentera sitt spel globalt. Vidare, tack vare integrationen av Epic Online Services (som

är en separat SDK), kan utvecklare på EGS utnyttja funktioner för plattformsoberoende multiplayer, topplistor, molnsparning m.m., inte bara på PC utan även i spel som släpps över flera plattformar. Det är dock frivilligt; EGS ställer endast kraven som listas ovan, men tillhandahåller verktyg för att uppfylla dem.

Lansering och uppföljning på EGS: När ett spel väl godkänts och lanserats på Epic Games Store, blir det synligt för alla användare via butikens gränssnitt. EGS har inte lika komplexa discovery-algoritmer som Steam, men nya spel visas under sektionen *New Releases* och kan lyftas fram på förstasidan genom redaktionellt urval eller om de trendar i försäljning. EGS saknar ett inbyggt önskelistesystem för kunder och har i skrivande stund färre communityfunktioner än Steam, till exempel inga offentliga diskussionsforum per spel. Därför kan marknadsföringen utanför plattformen och användning av externa kanaler vara viktig för att driva trafik till en EGS sida. Epic har dock kört aggressiva kampanjer för att dra in användare, till exempel gratis speltitlar varje vecka vilket har ökat EGS användarbas markant (Gdevelop, n.d.). Som utvecklare på EGS är det också möjligt att delta i Epic Games Stores egna rea evenemang och kampanjer, men dessa är mer kuraterade av Epic jämfört med Steams självbetjänings rea. Sammanfattningsvis får en indieutvecklare på EGS en större del av kakan och tillgång till ett växande ekosystem, men måste å andra sidan säkerställa att spelet lever upp till Epics krav, särskilt vad gäller crossplay och kvalitet, för att bli listat på plattformen.

6.3 Early Access och full lansering

Att välja mellan Early Access och en fullständig lansering är ett kritiskt beslut som påverkar hela spelets livscykel, från utveckling till marknadsföring och mottagande. Modellen man väljer avgör hur spelet finansieras, testas, formas av användare och slutligen tas emot av marknaden. I följande avsnitt behandlas båda lanseringsstrategierna, deras möjligheter och risker, samt vad som avgör när ett spel är redo att lämna Early Access.

6.3.1 Early Access: Möjligheter och risker

Early Access innebär att ett spel görs tillgängligt för försäljning medan det fortfarande är under utveckling, med ett löfte om framtida uppdateringar. Steam definierar modellen som ett sätt att låta spelare delta i utvecklingen genom att köpa en ofärdig version och bidra med

feedback (Steamworks, n.d.-a). Det är dock inte att likställa med förhandsbokning eller crowdfunding – Valve betonar att utvecklaren ska vara beredd att fullfölja projektet även utan försäljningsintäkter.

Bland de främsta fördelarna återfinns tidig intäktsgenerering. För små indieutvecklare kan försäljningen i Early Access vara avgörande för att slutföra projektet utan att söka stöd från investerare eller finansiärer (Frostilyte, 2024; Wayline, 2025a). Enligt en analys från Wayline (Wayline, 2025a) rapporterar studior som använder Early Access ofta högre intäkter än de som förlitar sig på traditionell lansering. Modellen gör det också möjligt att iterera baserat på konkret spelardata, där spelarna fungerar som ett utökat QA-team som upptäcker buggar, föreslår förbättringar och testar nya funktioner (Xsolla, 2021a).

Ytterligare en fördel är exponering. Early Access kan agera som en förlängd marknadsföringsperiod där spelet gradvis bygger upp en publik. Många spel listas på önskelistor, diskuteras i forum och får uppmärksamhet på Twitch och YouTube långt innan sin slutliga lansering. Exempel som *Besiege* (Spiderling Studios, 2015) och *Slay the Spire* (Mega Crit, 2017) visar hur långsiktigt community-engagemang kan leda till framgång (Bycer, 2021).

Den öppna utvecklingsmodellen ger också flexibilitet. Utvecklaren kan snabbt justera kursen utifrån feedback, lägga till nytt innehåll och anpassa spelets inriktning efter verkliga användardata (Smiley, 2019). Detta förutsätter dock att studion har en tydlig utvecklingsplan och kapacitet att filtrera och bearbeta input på ett effektivt sätt.

Men med friheten kommer risker. Ett ofärdigt första intryck kan leda till negativa recensioner som kvarstår permanent. Early Access lider av ett ihållande rykte för låg kvalitet och ofullbordade projekt, vilket skapar skepsis bland vissa konsumenter (Frostilyte, 2024). Modellen är också utsatt för så kallad funktionskryp – ett tillstånd där återkommande krav på nya funktioner gör att spelets omfattning växer ohanterligt (Wayline, 2025a).

Hanteringen av spelarens förväntningar är en annan utmaning. Regelbundna uppdateringar och öppen kommunikation är avgörande, och missköts dessa riskerar utvecklaren att tappa

både spelarnas engagemang och sitt eget momentum. Förtroendet är centralt i Early Access, och dess förlust kan försvåra eller omöjliggöra en framgångsrik lansering (Xsolla, 2021a).

Slutligen är risken för utebliven fullföljning påtaglig. Statistiskt når mer än hälften av alla spel som lanseras i Early Access aldrig en färdig 1.0-version, ofta på grund av finansiella svårigheter eller bristande planering (Wayline, 2025a). Utvecklare som överväger modellen bör därför göra en realistisk analys av sina resurser, ha en tydlig utvecklingsplan och vara beredda att stödja spelet långsiktigt – även vid motgångar.

6.3.2 Full lansering

Att välja en fullständig lansering utan föregående Early Access innebär att spelet släpps först när det betraktas som färdigt. Denna strategi används ofta för berättelsedrivna eller innehållsmässigt kompletta spel där upplevelsen ska konsumeras i ett sammanhang utan avbrott.

Fördelarna med en traditionell full lansering är tydliga. Utvecklaren har kontroll över det första intrycket och kan lansera ett polerat, stabilt och komplett spel. Det blir enklare att samordna pressutskick, recensionsexemplar, lanseringsevenemang och marknadsföringskampanjer kring ett specifikt datum, vilket ofta ger större genomslag (Smiley, 2019). Dessutom slipper utvecklaren pressen att leverera kontinuerliga uppdateringar under en pågående utvecklingsfas inför publik.

Samtidigt medför modellen vissa risker. Utvecklingen måste finansieras i sin helhet innan spelet börjar generera intäkter. Det innebär att utvecklaren måste klara hela produktionen utan den ekonomiska buffert som Early Access kan ge. Man går också miste om möjligheten att samla in spelardata och justera spelet innan release, vilket ökar pressen på interna tester och kvalitetssäkring.

Slutligen finns det marknadsföringsmässiga utmaningar. Utan den förlängda uppmärksamheten som en Early Access-period kan skapa, krävs det ofta mer resurser för att bygga förväntan inför lansering. Därför lämpar sig modellen bäst för spel som kan göra starkt intryck vid första kontakt – ofta genom en tydlig narrativ kärna eller unik visuell identitet – och där utvecklaren kan garantera att leverera ett konsekvent, färdigt paket vid släpp.

6.3.3 Lämpliga speltyper för Early Access

Early Access-modellen passar inte alla spel. Den är särskilt effektiv för genrer med hög återspelningsbarhet och systemfokus, där spelare kan engagera sig över tid och där nya funktioner successivt bygger vidare på en stabil kärna. Exempelvis passar genrer som roguelikes, överlevnadsspel och simulerings- eller byggspel särskilt väl (Frostilyte, 2024; Smiley, 2019).

Titlar som *Slay the Spire* och *Subnautica* (Unknown Worlds Entertainment, 2014) lyckades tidigt etablera spelbara loopar som både engagerade spelare och gav utvecklarna ett flöde av konkret återkoppling. Dessa spel kunde också itereras innehållsmässigt utan att förlora sin kärnupplevelse, vilket gjorde dem lämpliga för en stegvis utveckling i samverkan med publiken.

Däremot lämpar sig inte storydrivna, linjära spel lika väl. Här finns risken att viktiga berättelseelement avslöjas i förtid, vilket kan minska upplevelsen vid full lansering. Många utvecklare av sådana titlar föredrar därför att arbeta med demoversioner, stängda betor eller att helt enkelt vänta med släppet tills spelet är komplett (Xsolla, 2021a).

Frågan om när ett spel bör lämna Early Access är minst lika viktigt som när det bör gå in i det. Valve rekommenderar att man först gör detta när spelet uppfyller samtliga utlovade funktioner, håller hög stabilitet och erbjuder en fullständig spelupplevelse (Steamworks, n.d.-a). För tidig lansering kan skada spelets rykte och försämra mottagandet, medan en alltför utdragen Early Access-period riskerar att trötta ut publiken. En balans mellan teknisk färdigställande och marknadsstrategisk timing är därför avgörande.

6.3.4 Exempel på Early Access

Flera av de mest framgångsrika indieprojekten under det senaste decenniet har använt Early Access som en strategisk väg till lansering. Ett exempel är *Hades* (Supergiant Games, 2020), som under sin Early Access-period lyckades bygga en lojal spelarbas genom regelbundna uppdateringar, transparent utvecklingskommunikation och genomarbetad speldesign. Spelets 1.0-release hyllades både kritiskt och kommersiellt – ett direkt resultat av att Supergiant Games använt feedbacksystemet effektivt för att förfina spelets struktur (Smiley, 2019).

Även *Subnautica* illustrerar Early Access-modellens potential: det vann stor publik med sin atmosfär och överlevnadsmekanik långt innan 1.0-versionen släpptes. Utvecklarna använde regelbundna uppdateringar och öppen kommunikation för att förvalta och växa sin spelarbas, vilket i slutändan resulterade i både kritisk och kommersiell framgång.

Samtidigt finns tydliga exempel på projekt där Early Access visade sig problematiskt. Spelet *DayZ* (Bohemia Interactive, 2018) lanserades i Early Access 2013, men möttes av kraftig kritik under flera år för uteblivna uppdateringar och brist på transparens. Trots att det slutligen släpptes i full version efter fem år, hade mycket av spelarens förtroende redan gått förlorat – en konsekvens av bristande communityhantering och långa utvecklingspauser (Wayline, 2025a).

Statistik visar att en betydande andel av spelen som går in i Early Access aldrig når version 1.0. Detta sker ofta på grund av bristande projektledning, utebliven finansiering eller förlorad motivation. Även om vissa av dessa projekt kan ha haft potential, understryker dessa fall vikten av tydlig målsättning, realistisk planering och kontinuerlig kommunikation med spelarna (Xsolla, 2021a).

Dessa exempel visar att Early Access kan fungera som en kraftfull katalysator för både kvalitetsutveckling och community-byggande – men också att det är en modell med hög risk där framgång kräver disciplin, ärlighet och långsiktig uthållighet.

7. Diskussion

I det här examensarbetet har vi undersökt vad som krävs för att utveckla och lansera ett indiespel, med fokus på både tekniska och affärsmässiga aspekter. Vårt syfte var att skapa en grundläggande förståelse för hela processen – från idé till lansering – och att identifiera vanliga utmaningar samt strategier för att hantera dem.

Utifrån vår problemformulering ville vi ta reda på vilka delar en indieutvecklare behöver behärska för att skapa ett spel, samt hur val av plattformar och marknadsföringsstrategier påverkar ett spels framgång.

Vi har under arbetets gång uppmärksammat att spelutveckling som indiestudio är ett brett och krävande arbete som innefattar allt från programmering, grafik och ljuddesign till projektledning, kommunikation och affärstänk. Verktyg som Unreal Engine och Perforce underlättar det tekniska arbetet, men det krävs också god planering, tydlig ansvarsfördelning och realistiska mål för att projektet ska gå att genomföra.

Vi har också insett att marknadsföring och synlighet är minst lika viktigt som själva spelet. Att bygga upp ett intresse tidigt, till exempel via sociala medier, devlogs eller en community, kan vara avgörande för hur väl spelet tas emot. Early Access är ett kraftfullt verktyg för både testning och finansiering, men det kräver att man aktivt lyssnar på feedback och upprätthåller en relation med spelarna.

Att arbeta med det här examensarbetet har gett oss en tydlig bild av hur komplext och omfattande spelutveckling är – även i liten skala. Vi har fått erfarenhet av samarbete, problemlösning och projektledning, men också lärt oss att kreativitet måste balanseras mot tekniska och praktiska begränsningar. Det har blivit tydligt för oss att spelutveckling inte bara handlar om att skapa ett spel – det handlar om att bygga ett projekt från grunden och ta ansvar för hela dess livscykel.

7.1 Förbättringsområden

Under arbetets gång har vi identifierat flera områden som inte behandlats i detalj men som är relevanta att fördjupa sig i vid framtida spelprojekt. Ett centralt förbättringsområde är det

fortsatta arbetet efter lansering – särskilt när det gäller att ta tillvara på spelarfeedback genom testning, enkäter och analys av spelbeteende. Att implementera system för speltelemetri kan ge värdefulla insikter som används för att förbättra spelets balans, användarupplevelse och innehåll.

Vi har också reflekterat över valet av verktyg. Unreal Engine och Perforce har fungerat väl för vårt projekt, men i framtiden kan det vara relevant att utvärdera alternativa spelmotorer som Unity eller Godot, som erbjuder andra arbetsflöden och kan vara mer resurseffektiva beroende på speltyp. För versionshantering kan Git i kombination med plattformar som GitHub eller GitLab vara ett mer lättillgängligt alternativ, särskilt för mindre team.

Slutligen hade det varit värdefullt att gå djupare in på olika affärsmodeller som inte har behandlats och finansieringsstrategier. Exempelvis DLC eller free-to-play-modeller kan påverka både utvecklingsförloppet och spelets framgång på marknaden. En mer ingående analys av dessa alternativ skulle ha kunnat stärka det affärsmässiga perspektivet i vårt arbete ytterligare.

7.2 Slutsats

Sammanfattningsvis har vi besvarat våra frågeställningar och fått en konkret förståelse för både möjligheter och utmaningar inom indiespelutveckling. Den kunskap och erfarenhet vi har fått kommer att vara värdefull i framtiden, oavsett om vi fortsätter med spelutveckling eller andra områden inom IT.

Källförteckning

BackerKit. (2021, June 8). *The 3 stages of playtesting — Internal, Local, and Blind*. BackerKit. https://www.backerkit.com/blog/tabletop-games-crowdfunding-roadmap/playtest/the-3-stages-of-playtesting-internal-local-and-blind/

Belwariar, R. (2016, June 16). *A* Search Algorithm*. GeeksforGeeks. https://www.geeksforgeeks.org/a-search-algorithm/

Bramble, R. (2023a, January 30). *14 Tips For Generating Best-Selling Video Games Ideas*. https://gamemaker.io/en/blog/video-game-ideas

Bramble, R. (2023b, February 13). *Experts Explain How To Run A Video Game Crowdfund Campaign*. https://gamemaker.io/en/blog/how-to-run-a-successful-crowdfunding-campaign

Bycer, J. (2021, January 13). *The Successful Steps of Early Access Games*. Game Developer. https://www.gamedeveloper.com/design/the-successful-steps-of-early-access-games

Caoili, E. (2008, May 1). *Feature: "Building a Mindset for Rapid Iteration Part 1."* Game Developer. https://www.gamedeveloper.com/game-platforms/feature-building-a-mindset-for-rapid-iteration-part-1-

Carson, O. (2024, August 15). Comparing Popular Game Engines. *PubNub*. https://www.pubnub.com/blog/comparing-popular-game-engines/

Chichmanov, G. (2024, April 24). *Buy asset pack vs. custom game models: A guide to Indie Game development*. https://www.comintedlabs.io/news/buy-asset-pack-vs-custom-game-models-a-guide-to-indie-game-development

Clarke, S. (2023, October 5). *Testing Games with AI Bots*. Game Developer. https://www.gamedeveloper.com/programming/testing-games-with-ai-bots

Corsica, C. (2020, December 15). Week 9: Profiling and Optimizing. *Technical Animator*. https://www.christiancorsica.com/post/week-9-profiling-and-optimizing

Cowley, V. D. (2015, July 30). *Unreal Developers at GDC Europe and Gamescom*. Unreal Engine. https://www.unrealengine.com/blog/unreal-developers-at-gdc-europe-and-gamescom

Dankoff, J. (2014, March 20). *Game Telemetry with DNA Tracking on Assassin's Creed*. Game Developer. https://www.gamedeveloper.com/design/game-telemetry-with-dna-tracking-on-assassin-s-creed

Denby, L. (2022, December 15). How many Steam wishlists does my indie game need? - Game If You Are. *Game If You Are - The Indie Game Marketing Agency*. https://gameifyouare.com/2022/12/15/how-many-steam-wishlists-does-my-indie-game-need

Epic Developer Community Forums. (2016, March 2). *Licence Terms*. Epic Developer Community Forums. https://forums.unrealengine.com/t/licence-terms/57064

Epic Games Developer. (n.d.-a). *AI Debugging*. Unreal Engine Developer Documentation. Retrieved May 2, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/ai-debugging-in-unreal-engine

Epic Games Developer. (n.d.-b). *Artificial Intelligence*. Unreal Engine Developer Documentation. Retrieved March 18, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/artificial-intelligence-in-unreal-engine

Epic Games Developer. (n.d.-c). *Behavior Tree Overview*. Unreal Engine Developer Documentation. Retrieved April 15, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/behavior-tree-in-unreal-engine---overview

Epic Games Developer. (n.d.-d). *Greyboxing*. Unreal Engine Developer Documentation. Retrieved March 20, 2025, from https://dev.epicgames.com/documentation/en-us/uefn/greyboxing-in-unreal-editor-for-fortnite

Epic Games Developer. (n.d.-e). *Networking Overview*. Unreal Engine Developer Documentation. Retrieved March 20, 2025, from

https://dev.epicgames.com/documentation/en-us/unreal-engine/networking-overview-for-unreal-engine

Epic Games Developer. (n.d.-f). *Simple versus Complex Collision*. Unreal Engine Developer Documentation. Retrieved April 21, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/simple-versus-complex-collision-in-unreal-engine

Epic Games Developer. (n.d.-g). *Stat Commands*. Unreal Engine Developer Documentation. Retrieved April 21, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/stat-commands-in-unreal-engine

Epic Games Developer. (n.d.-h). *Unreal Engine 5.5 Documentation*. Unreal Engine Developer Documentation. Retrieved April 11, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/unreal-engine-5-5-documentation

Epic Games Developer. (n.d.-i). *Unreal Insights*. Unreal Engine Developer Documentation. Retrieved April 21, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/unreal-insights-in-unreal-engine

Frostilyte. (2024, August 19). *Steam Early Access: Is It Time to Change My Stance?* Frostilyte Writes. https://frostilyte.ca/2024/08/19/steam-early-access-is-it-time-to-change-my-stance

Game Development Stack Exchange. (2024, January 22). *Do real game developers use blueprints, or do they always use C++?* Game Development Stack Exchange. https://gamedev.stackexchange.com/questions/208684/do-real-game-developers-use-blueprints-or-do-they-always-use-c

Games, E. (2023, March 9). *Epic Games Store Launches Self-Publishing Tools for Game Developers and Publishers*. https://store.epicgames.com/en-US/news/epic-games-store-launches-self-publishing-tools-for-game-developers-and-publishers

Games, H. (2023, July 10). *A Simple Guide to Steam Direct: How to Distribute Your Game on Steam*.

GameMakerBlog | Mastering GameMaker Studio: Reviews, Tutorials, and Indie Game Insights; Hikati Games LLC. https://gamemakerblog.com/2023/07/10/a-simple-guide-to-steam-direct-how-to-distribute-your-game-on-steam/

Gdevelop. (n.d.). *How to Publish Your Game on the Epic Games Store in 2024 (And Why You Should)*. Gdevelop. Retrieved April 29, 2025, from https://gdevelop.io/page/how-to-publish-game-on-epic-games-store-and-why-you-should

Gitlab. (2022, February 4). *What is version control?* GItlab; GitLab. https://about.gitlab.com/topics/version-control/

Golding, N. F., Hu, L., & Ge Lin, C. (2022a). *Some Models are Useful Sometimes: And Other Lessons for Indie Game Marketing from Action Research*. CBS Research Portal. https://research.cbs.dk/en/studentProjects/some-models-are-useful-sometimes-and-other-lessons-for-indie-game

Golding, N. F., Hu, L., & Ge Lin, C. (2022b). *Some Models are Useful Sometimes: And Other Lessons for Indie Game Marketing from Action Research*. https://research.cbs.dk/en/studentProjects/some-models-are-useful-sometimes-and-other-lessons-for-indie-game

Gordon, L. (2021, February 15). How the nitty gritty of video game funding shapes what you play. *The Guardian*. https://www.theguardian.com/games/2021/feb/15/video-game-funding-indie-investment

GPUopen. (2023, December 14). *Unreal Engine performance guide*. AMD GPUOpen; GPUOpen by AMD. https://gpuopen.com/learn/unreal-engine-performance-guide/

Guay, J.-S. (2024, May 15). *World Building Guide*. Epic Games Developer. https://dev.epicgames.com/community/learning/knowledge-base/r6wl/unreal-engine-world-building-guide

Handrahan, M. (2020, April 24). *Valve's advice for making your game thrive after launch*.

GamesIndustry.biz. https://www.gamesindustry.biz/valves-advice-to-keep-your-game-thriving-after-launch

Isaksson, J. (2024). *Comparative Analysis of 3dcave Generation Methods for Procedural Environments*. https://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-229356

iXie. (2023, May 26). *7 Major Challenges faced by Indie Game Developers*. iXie Gaming. https://www.ixiegaming.com/blog/challenges-faced-by-indie-game-developers/

Kahn, S. (2019, November 26). *Perforce Source Control for game development*. Kahncode; Samuel Kahn. https://kahncode.com/2019/11/26/perforce-source-control-for-game-development/

Kultima, A. (2015, September 22). Developers' perspectives on iteration in game development. *Proceedings of the 19th International Academic Mindtrek Conference*. AcademicMindTrek'15: Academic Mindtrek Conference 2015, Tampere Finland. https://doi.org/10.1145/2818187.2818298

Larsson, B. (n.d.). *UnrealEngine-ObjectPool: Unreal engine plugin providing an object pool packaged in a world subsystem for easy use*. Github. Retrieved April 21, 2025, from https://github.com/bennystarfighter/UnrealEngine-ObjectPool

Lin, D., Bezemer, C.-P., & Hassan, A. E. (2018). An empirical study of early access games on the Steam platform. *Empirical Software Engineer*, *23*(2), 771–799.

Looman, T. (2022, November 3). *Unreal Engine Game Optimization on a Budget*. Tom Looman. https://www.tomlooman.com/unrealengine-optimization-talk/

Lovato, N. (2017, February 15). *Marketing Your Indie Game On A Zero-Dollar Budget — GameAnalytics*. https://www.gameanalytics.com/blog/marketing-indie-game-without-budget

Mittal, S. (2024, August 31). *What is Grey Boxing? and why is it so good! - Suryanshu Mittal*. Medium. https://medium.com/@alex.underworldent.ltd/what-is-grey-boxing-and-why-is-it-so-good-4b2674f1e003

Mullich, D. (2015, February 3). *Roles On A Game Development Team*. David Mullich.

https://davidmullich.com/2015/02/02/roles-on-a-game-development-team/

Nadiy. (2024, September 20). *AI in Gaming: How AI is Used to Create Intelligent Game Characters and Opponents*. https://www.lizard.global/blog/ai-in-gaming-how-ai-is-used-to-create-intelligent-game-characters-opponents

Obedkov, E. (2023, March 9). *Epic Games Store introduces self-publishing tools with support for IARC age rating system*. Game World Observer. https://gameworldobserver.com/2023/03/09/epic-games-store-self-publishing-tools

OpenGameArt. (2011, July 14). *FAQ*. OpenGameArt.org. https://opengameart.org/content/faq

Pajkovic, N. (2023, April 25). *What is an Indie Game? A Comprehensive Guide*. Toronto Film School. https://www.torontofilmschool.ca/blog/what-is-an-indie-game/

Perforce. (n.d.). *Documentation*. Retrieved April 11, 2025, from https://www.perforce.com/support/self-service-resources/documentation

Pilakowski, M. (2017, May 23). *Game Design in College Composition? Yep*. Medium. https://medium.com/@mpilakow/game-design-in-college-composition-yep-ba4f07de1e57

Plugova, S. (2021). *Visual marketing strategy usage for indie game promotion*. https://www.eek.ee/download.php?t=kb&dok=p1f4phd5h6ke91uks17v9jp81ckf3.pdf

Ponce, T. (2016, July 6). *The heartbreaking saga of Mighty No. 9*. Destructoid. https://www.destructoid.com/the-heartbreaking-saga-of-mighty-no-9/

Powell, R. (2016). *Positive and Negative effects of Game Analytics in the Game Design process : A Grounded Theory Study*. https://www.uppsatser.se/uppsats/7229e83539/

Sharman, A. (2024, October 27). *Steam Next Fest: An Analysis - Adam Sharman*. Medium. https://medium.com/@ThatOneAJ/steam-next-fest-an-analysis-e8117db0cbaa

Smiley, D. (2019, October 17). *Should your game be released on early access?* Game Developer. https://www.gamedeveloper.com/business/should-your-game-be-released-on-early-access-

Steamworks. (n.d.-a). *Early Access (Steamworks Documentation)*. Steamworks. Retrieved April 29,

2025, from https://partner.steamgames.com/doc/store/earlyaccess

Steamworks. (n.d.-b). *Stats and Achievements (Steamworks Documentation)*. Retrieved April 29, 2025, from https://partner.steamgames.com/doc/features/achievements

Steamworks. (n.d.-c). *Steam Cloud (Steamworks Documentation)*. Retrieved April 29, 2025, from https://partner.steamgames.com/doc/features/cloud

Steamworks. (n.d.-d). *Update Visibility Rounds (Steamworks Documentation)*. Retrieved April 21, 2025, from https://partner.steamgames.com/doc/marketing/visibility/update_rounds

Sztajer, P. (2011, August 2). *Playtesting 103: What to Measure*. Game Developer. https://www.gamedeveloper.com/design/playtesting-103-what-to-measure

Vblanco. (2022, February 24). *Expert's guide to unreal engine performance*. Epic Games Developer. https://dev.epicgames.com/community/learning/tutorials/3o6/expert-s-guide-to-unreal-engine-performance

Verón, J., Pérez, C., Calero, C., Moraga, M., Pérez, F., & Cetina, C. (2024). A comparative analysis of energy consumption between visual scripting models and C++ in unreal engine: Raising awareness on the importance of green MDD. *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*, 114–125.

Wayline. (2025a, February 25). *Early Access: Indie Savior or Slow Death?* https://www.wayline.io/blog/early-access-indie-savior-or-slow-death

Wayline. (2025b, March 14). *Solo or Team Development? The Honest Pros and Cons*. Wayline. https://www.wayline.io/blog/solo-or-team-development-the-honest-pros-and-cons

Wikipedia contributors. (2025a, April 27). *Perlin noise*. Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Perlin_noise

Wikipedia contributors. (2025b, May 2). *Non-player character*. Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Non-player_character

Wu, M. (2017, December). *Case study the indie game industry and help indie developers achieve their success in digital marketing*. https://repository.library.northeastern.edu/files/neu:cj82rk43p

Xsolla. (2021a, October 11). *Early Access: Will It Help Or Hurt Your Game?* https://xsolla.com/blog/early-access-will-it-help-or-hurt-your-game

Xsolla. (2021b, October 11). *Self-Publish On Steam: The Ultimate Guide*. Xsolla. https://xsolla.com/blog/self-publish-on-steam-the-ultimate-guide

Yasha_eix. (2023, April 29). *The Key Differences between AAA and Indie Game Studios*. Zeka Design. https://www.zekagraphic.com/aaa-vs-indie-game-studios/

Zak, R. (2020, December 26). *After 2 years, the Epic Games Store is still a golden ticket for devs and irresistible bait for gamers*. PC Gamer. https://www.pcgamer.com/after-2-years-the-epic-games-store-is-still-a-golden-ticket-for-devs-and-irresistible-bait-for-gamers/

Zohrabyan, S. (2023, December 27). *Indiegogo and Kickstarter Statistics: 1st Quarter of 2023*. Crowdfunding PR & Marketing Blog - The Crowdfunding Formula. https://blog.thecrowdfundingformula.com/indiegogo-and-kickstarter-stats/

Zolotarenko, J. (2024, March 11). *Indie, AA, and AAA Games: The Ultimate Guide - Julia Zolotarenko*. Medium. https://hitberrygames.medium.com/indie-aa-and-aaa-games-the-ultimate-guide-4147c393ca72

Zucconi, A. (2022, June 5). *The World Generation of Minecraft*. Alan Zucconi. https://www.alanzucconi.com/2022/06/05/minecraft-world-generation/

Zukalous. (2021, December 27). *What worked in 2021? A summary of game dev postmortems*. https://howtomarketagame.com/2021/12/27/what-worked-in-2021/