Tallinna Ülikool

Digitehnoloogiate Instituut

Mängureeglite balansseerimine

Seminaritöö

Autor: Alari Alev

Juhendaja: Martin Sillaots

Tallinn 2016

Sisukord

Sissejuhatus

Videomänge on aina lihtsam igaühe poolt luua, kuid tihti ei panda rõhku olulisematele aspektidele videomängu disainis. Sellest tulenevalt lastakse välja kehvalt loodud mänge, mis tekitavad mängijates ainult pahameelt. Seega on oluline rõhutada mängureeglite arendamise ning rakendamise protsessi olulisust ja anda suunitlusi tulevastele arendajatele. Samuti on ka vähe eestikeelseid ülevaateid antud teemast. Mida rohkem materjale mänguarendajatel on, seda paremaid mänge tulevikus luuakse.

Töö keskendub mängureeglite balansseerimisele videomängudes. Seletatakse lahti mõiste mängureeglid ning kuidas need mõjutavad terviklikku mänguarenduse protsessi. Kui mängureeglid on loodud, siis peab neid hakkama balansseerima. Reeglid peavad looma õhkkonna, kus on vaja käigupealt erinevad strateegiad välja mõelda vastase alistamiseks. Välja ei tohiks kujuneda kindlad matemaatiliselt välja töötatud ainuõiged lahendused. Näidetena kasutatakse populaarsemaid frantsiise erinevatest mängužanritest, et oleks võimalikult lai skoop kaetud. Töö käigus kujunevad välja soovitused tulevastele mänguarendajatele, mida tuleks jälgida mängureeglite loomise protsessis ning kuidas neid tasakaalustada.

Töö koosneb erinevatest peatükkideks, mille sidususeks on mängureeglid. Peatükkides tuuakse välja, kuidas valida rakendatavaid mängureegleid žanri põhiselt ning protsesse, kuidas neid võiks saada rakendada enda loodavas mängus. Kõige olulisem on loomulikult balansseerimise osa, seega saab tutvustada protsesse, kuidas ja miks peab rakendatavaid reegleid balansseerima. Samuti tutvustatakse ka erinevaid tehnikaid ja testimise võimalusi, kuidas mängureegleid võrdsustada.

1. Mängureeglid

Videomängude reeglid on osa mängudisainist. Need määravad ära tegevused, mida mängija teha saab või ei saa. Kui mängida malet mitte digitaalsel kujul, siis on võimalik nuppe liigutada nii, kuidas mängija seda soovib. Digitaalsel kujul aga määravad reeglid täpsed kohad, kuhu nuppe liigutada saab (Hobby Gamedev, 2011). Mängude kulg toimub kindla raamistiku põhjal, mis on paika pandud reeglitega. Reeglid määravad selle, kuidas mingi objekt mängumaailmas reageerib mängija tegevustele ning seeläbi juhtub mängus midagi, mis on reeglite poolt määratud. Samuti määravad reeglid selle, kuidas mängija saab mängu võita. Seega on videomängude reeglid ühed olulisemad komponendid videomängude disainis (Gingold, 2003).

Reegleid tuleb tasakaalustada objektide vahel. Kui arendaja hakkab mängu looma, siis ta loob kõigepealt tühja staatilise maailma. See maailm tuleb täita objektidega ning need objektid jagunevad kaheks. Staatilised objektid ei tee maailmas midagi. Näiteks pildid seintel või trepiastmed. Dünaamilised objektid on need, mis teevad midagi. Nendeks võivad olla tegelased, uks mis avaneb või eelneva male näite puhul oleks selleks objektiks malenupp (Make-Video-Games.com, kuupäev puudub). Just neid dünaamilisi objekte on vaja hakata balansseerima. Neile tuleb määrata sündmused ehk mida objektid teha saavad. Olgu selleks hüppamine, ujumine, avanemine, liikumine jne. Sündmused määravad ka selle, kuidas erinevad objektid hakkavad suhtlema omavahel. Välja on kujunenud kahte tüüpi reegleid. *Paidea* reeglid ning *ludus* reeglid (Frasca, 2003).

1.1. *Paidea* ja *ludus* reeglid

Paidea on kreekakeelne sõna, mis tähendab nii last kui ka kooli. *Ludus* on ladinakeelne sõna, mis tähendab mängu. *Paidea* reeglid on loodud selleks, et mängu üldse mängida ning *ludus* reeglid on vajalikud võitmise või kaotamise jaoks. Näiteks males kirjeldavad *paidea* reeglid, kuidas iga malenupp liigub ning *ludus* reeglid määravad lõpuoleku selleks, et näidata, kes võitis ja kes kaotas (Juul, 2001).

Näitena vaatleme avatud linnaehituse simulaatorit *SimCity*. *Paidea* tüüpi mäng, kus ei ole otseseid võidu või kaotuse reegleid määratud. *Paidea* tüüpi mängud ongi enamasti liivakasti (ingl *sandbox*) stiilis mängud, kus otseseid võidutingimusi ei ole. Mängija

saab vabalt maailmas ringi liigelda. Kui aga mängija ise otsustab ehitada linna, kus on näiteks 10000 elanikku, siis nad liiguvad *ludus* tegevuse poole, kus mängija on ise endale määranud võidutingimuse. Kui vaadelda mängu *Tetris*, siis tegemist on *ludus* tüüpi mänguga, millel on kindel eesmärk, mida mängija peab täitma. Kui ta selle nõudmise täidab edukalt, siis ta on mängu võitnud ning talle avaneb uus raskem tase (Tabel 1).

Tabel 1. *Paidea* reeglid ja *ludus* reeglid

	Paidea reegli näidis	*Ludus* reegli näidis
SimCity (*Paidea* tüüpi mäng)	Kui kuritegevus tase on liiga kõrge, siis linna populatsioon langeb.	Puudub
Tetris (*Ludus* tüüpi mäng)	Kui plokid moodustavad tervikliku taseme, siis see tase kaob ära ning mängija teenib punkte.	Hoida plokkide tase võimalikult madalal.

Susana Tosca (2003) uuris videomängude poolt edastatud ülesandeid mängijale ning avastas, et välja on kujunenud nii-öelda tugevad ja pehmed reeglid. Tugevad reeglid määravad selle, kuidas maailm reageerib mängija tegevustele. Näiteks objektide omadused, käitumised, mängu dünaamika ning ka mängu lõppeesmärgi. Pehmed reeglid on selleks välja kujunenud, et rakendada neid tugevaid reegleid väikeste tükkidena, mida mängija saab omavoliliselt rakendada. Näiteks videomängus *Super Mario Bros 3* on tugevaks reegliks Mario käitumine. Mario saab hüpata, joosta ning ujuda. Pehmed reeglid on tegevuste jada, mida Mario peab läbi viima, et muutuda surematuks ehk lõhkuma telliskivi altpoolt ning puutuma tähte, mis tuleb sellest kivist välja. Sarnaselt *Super Mario Bros 3* mängule on ka eelnevalt vaadeldud mängudes oma pehmed ja tugevad reeglid (Tabel 2).

Tabel 2. Pehmed ja tugevad reeglid

	Pehme reegli näidis	Tugeva reegli näidis
Super Mario Bors 3	Lõhu telliskivi ning püüa täht kinni	Mario saab hüpata, joosta ning ujuda
SimCity	Mängija ehitab politsei jaoskonna ning kuritegevus langeb seeläbi	Mängija saab ehitada maju
Tetris	Plokk tekib ekraani üleval ääres ning mängija kukutab selle alla sobivasse kohta	Mängija saab laduda plokke üksteise peale

1.2. Sümboolsed ja semantilised *paidea* reeglid

Kui rakendada pehmete ja tugevate reeglite kontseptsiooni *paidea* ja *ludus* reeglitel, siis saame need omakorda kategooriateks lahti lüüa. Alustades *paidea* reeglitest. *Paidea* reeglitest tuleks järgnevalt välja sümboolsed *paidea* reeglid ning semantilised *paidea* reeglid (Tabel 3). Sümboolsed *paidea* reeglid määravad selle, mida mängija saab või ei saa teha virtuaalses ruumis. Samuti see seob füüsilised interaktsioonid virtuaalsete interaktsioonidega (Ang, 2005). Semantilised *paidea* reeglid aga defineerivad interaktsioonide põhjused, ehk kuidas interaktsioonid on omavahel seotud ning millised on nende interaktsioonide tagajärjed mängumaailmale (Ang, 2005). Vaadeldes eelnevalt mainitud mänge, siis *Tetrises* määravad sümboolsed *paidea* reeglid selle, kuidas mängija saab manipuleerida igat plokki ning semantilised *paidea* reeglid määravad selle, kuidas plokid suhtlevad omavahel.

Tabel 3. Sümboolsed ja semantilised *paidea* reeglid

Paidea reeglid	Sümboolsed	Semantilised
Super Mario Bros 3	Vajuta paremat noole klahvi selleks, et liigutada Mariot ekraanil paremale	Mario puudutab tähte ning seeläbi muutub surematuks lühikeseks ajaks
Tetris	Vajuta alla noole klahvi ja plokk kukub alla	Kui plokid katavad terve taseme, siis kaob see tase ära

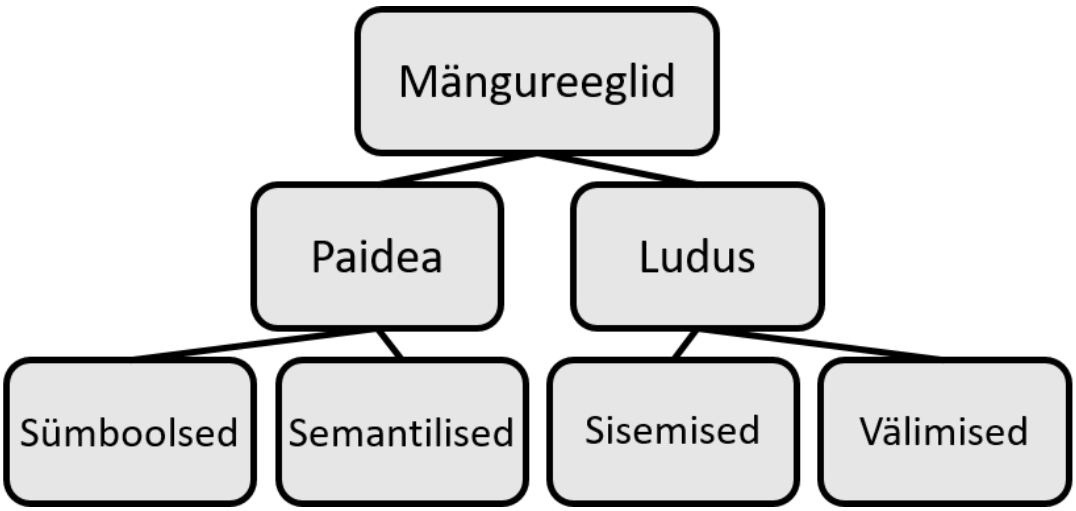
1.3. Sisemised ja välimised *ludus* reeglid

Kui vaadelda *ludus* reegleid, siis nendele saab rakendada sama kontseptsiooni. *Ludus* reegli kategooriad on sisemised *ludus* reeglid ning välimised *ludus* reeglid (Tabel 4). Sisemised *ludus* reeglid aitavad mängu võitmisele kaasa kaudselt ja välised *ludus* reeglid aitavad mängu võitmisele kaasa otseselt (Ang, 2005). Vaadeldes taas eelnevalt mainitud mängu *Super Mario Bros 3* saab väliseks *ludus* reegliks määrata üleüldise lõppeesmärgi ehk printsessi päästmise. Küll aga peab Mario oma teekonnal läbima erinevaid katseid ning ellu jääma. Ehk siis näiteks kilpkondade tapmine ning telliskividest seente püüdmine. Ainult sisemisi *ludus* reegleid täites ei saa mängija mängu võita, kuid need tegevused aitaksid saavutada lõppeesmärki.

Tabel 4. Sisemised ja välimised *ludus* reeglid

Ludus reeglid	Sisemised	Välimised
Super Mario Bros 3	Tapa kilpkonn	Päästa printsess
Tetris	Kasuta plokki	Hoia plokkide tase võimalikult madal

Reeglite kirjeldusest on näha, et erinevad tüübid on omavahel seotud hierarhiliselt (Joonis 1). Igale madalama taseme reeglile saab rakendada kõrgema taseme seadusi. Erinevused võivad olla väikesed, kuid need on siiski olemas ning mängureeglite loomisel peab mõtlema iga väiksema nüansi peale.



Joonis 1. Mängureeglite hierarhia (autori joonis)

Kui on selge, milline saab mäng olema, siis peab mõtlema, kuidas kavandada rakendatavaid reegleid enda loodavas mängus. Arendaja peab teadma, milliseid reeglite tüüpe tema kavandatavas mängus on kõige loogilisem ning kõige parem kasutada. Reeglite kavandamisest ning rakendamisest aga ei piisa. Need peavad samuti suhtlema üksteisega sidusalt ning seetõttu on vaja rakendatavaid reegleid pidevalt testida ning balansseerida.

2. Reeglite balansseerimine

Selles staadiumis on olemas seosed ja tingimused, kuidas asjad mängus käituda võiks, kuid puuduvad objektid ning sündmused, mis need seoks. Selge on see, milline saab mäng olema ning milliseid reegleid on võimalik kasutada ning kuidas kuvada kasutajale võimalusi, mida ta teha saab. Ühtlasi on ka teada, kuidas kasutaja mänguga suhtleb ning kuidas mäng suhtleb temaga. Küll on aga vaja balansseerida mängus kasutatavaid objekte selleks, et interaktsioonid mängumaailmaga järgiks kindlat raskusastet (ingl *difficulty curve*). Näiteks, kui anda mängu alguses mängijale kohe kõige võimsam relv, siis see eemaldab igasuguse pinge mängu edasimängimisel (Tvtropes, kuupäev puudub). Seetõttu tuleb teha palju testimist selles osas, kui tugevad mingid objektid on. Selline tasakaalustamine on kõige olulisem just *ludus* tüüpi mängudes, kus on paigas kindlad võidutingimused. Siiski ei tohiks tunduda, et mäng teeb sohki.

Sohi tegemise all mõeldakse olukorda, kus mängus olevad vastased, mis ei ole juhitud inimeste poolt, on liiga võimsad. Kui mängijal on ilmselge eelis, aga on näha, et vastane ikka suudab osutada märgatavat vastupanu või ta suudab luua objekte, mida ta enam teha ei tohiks, siis on ilmselgelt tegu sohiga (Tvtropes, kuupäev puudub). Näiteks võib tuua *Heroes of Might and Magic* mänguseeria, kus eesmärk mängijal on üle võtta kõik linnad, mis antud kaardil on ning seeläbi alistada vaenlane. Kui on olukord, kus vastasel on ainult üks linn järgi ning mängija kontrollib ülejäänud linnu, siis on tal ilmselge eelis olemas. Linnad loovad talle üksusi, mida saab lahingutes kasutada ning samuti genereeritakse seal ka sissetulekut, millega neid üksusi soetada. Kui aga vastane suudab ikkagi genereerida rohkem üksusi kui mängija, siis tegu on ilmselge sohitegemisega.

Kui arvuti poolt juhitavad vastased ei tee otseselt sohki, siis tuleb ikkagi balansseerida nende elu, mis objekte nad mängumaailmas kasutada saavad ning kui palju nad mängija poolt juhitud avatarile haiget teha võivad. Siinkohal võib anda mängijale võimaluse ka omalt poolt panustada balansseerimisse ehk anda võimalus valida tal raskusaste (Game Design Concepts, 2009). Madalamate raskusastmete puhul on vastastel vähem elusid ja nad teevad vähem haiget ning kõrgemate raskusastmete puhul vastupidi. Küll aga tuleb leida need piirid, mis määravad vastaste numbrilised väärtused nendes raskusastmetes.

Seetõttu on oluline kavandada rakendatavad reeglid hoolega läbi enne, kui jõutakse nende rakendamise etapini.

2.1. Reeglite kavandamine

Kui mänguarendaja hakkab kavandama reegleid oma mängu jaoks, siis ta peab olema kindlaks teinud, milline tema mäng saab olema. Milliseid reeglite tüüpe saab ta kasutama hakata ning millised saavad olema eesmärgid, mille poole mängijad pürgima hakkavad. Arendaja peab panema kirja palju küsimusi, millele ta suudaks ka vastata. Näiteks millist lugu ta soovib jutustada oma mängus. Mida ta soovib saavutada oma mänguga. Mida ta soovib, et mängijad tunneks mängu lõpus (wikiHow, kuupäev puudub). Need on vaid vähesed küsimused, millele arendaja peab vastuse leidma.

Enne kui saab aga mängu kavandamisega edasi minna, peab kavandama reeglid. Kuna reeglid võivad olla väga erinevad ja mitmekesised, siis nende loomine võib olla kõige frustratsiooni tekitavaim protsess. Ibrahim Yucel (2014) ütleb, et arendaja peab mõtlema kõikide objektide või ressurrsside peale, mida mängus kasutama hakatakse. Need tuleks arusaadaval viisil kirja panna, et hiljem oleks neid lihtsam rakendada. Igale objektile või ressurssile peab lisama ka juurde kirjelduse, mida ta teeb. Ehk parameetrid kuidas need käituda võiksid mängumaailmas. Kui tegu on mingi objektiga, mis puutub kokku teiste objektidega, siis tuleb arvestada ka sellega, kuidas see objekt suhtleb teiste objektidega. Parameetrite muutused, objektide vahelisel suhtlusel, peavad olema läbimõeldud. Halvasti rakendatud objektid või ressursid võivad tekitada olukorra, kus juba välja lastud mängule peab looma juurde uusi reegleid. Need aga võivad tekitada üleliigset segadust mängjaskonna seas, ning raskendada mängust arusaamist. Objektidele seatud käitumised peavad olema põhjendatud. Näiteks miks mingi objekt just nii käituma peab. See annab teada, kas see objekt on rakendatud õigesti, või see on üleliigne. Üleliigse olemise korral saab objekti ning sellega seotud reeglid eemaldada, muutes kavandamise protsessi lühemaks.

Kuna on olemas reeglid reeglite kavandamiseks, siis on soovitatav uurida olemasolevaid foorumeid või muid allikaid, kuidas mingi objekt käituda võiks. Näiteks ukse objektil võiks soovitatav käitumisviis olla, et see mängumaailmas avaneb ning sulgub ja laseb mängijal liikuda erinevate ruumide vahel. Reeglid peavad edastama selged

ühetähenduslikud juhtnöörid, kuidas mängu mängida. Kuigi mängu voolavus, kaasahaaravus ning lõbu on alati esmased eesmärgid, siis on oluline reeglitest arusaamine, et saavutada need eesmärgid (Chen, 2006).

Arendajad tihti loovad rida väiteid, mis moodustavad reeglid. Need väited on edastatud paragrahvidena, kus arendaja kirjeldab igat objekti või ressurssi ning määrab tingimused. Küll on aga neid kirjeldusi ning tingimusi keeruline kirja panna sellisel viisil, et see oleks arusaadav ning ühetähenduslik (Yucel, 2014). Et need saaks soovitud kujul kirja pandud, siis on välja kujunenud kolm viisi, mis lihtsustavad reeglite kavandamist.

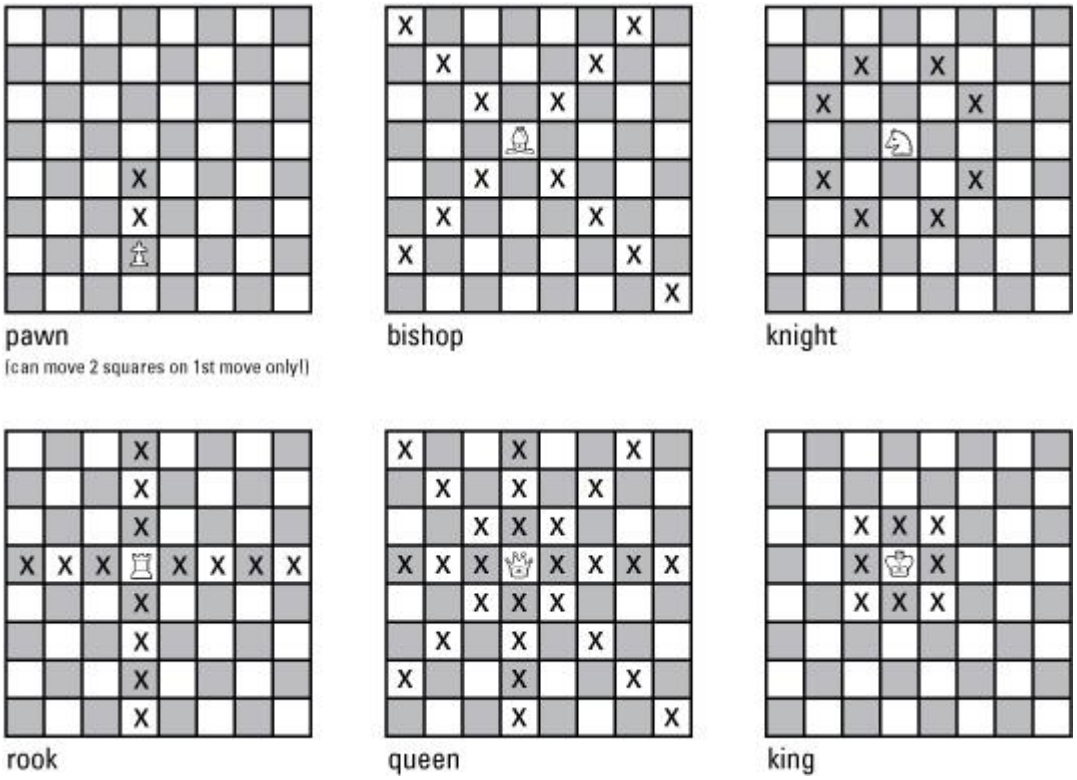
Liigendamine

Liigendamine (ingl *chunking*) on peamine ning lihtsaim viis kuidas keerulisi reeglite kogumikke organiseerida, meelde jätta ning refereerida. William Lidwell, Kritina Holden ja Jill Butler (2010) defineerivad liigendatud tükke kui väikseid informatsiooni kilde lühimälu jaoks. Ehk tähtede kombinatsioon, sõnad või numbrite jada. Maksimaalne arv tükke mida saab effektiivselt protsessida on neli. Sellest tulenevalt, kui on välja kujunenud keeruline reeglite jada, mingi tegevuse läbiviimiseks, siis võib need tegevused tükkideks jagada. Näiteks Dungeons and Dragons mängu puhul oleks esimeseks tükiks võitlus eelne faas, kus pannakse paika olekud, kes on vastastest teadlikud. Teine tükk oleks initsiatiivi määramise faas ehk kes esimesena ründab ning kolmandaks tükiks oleks võitlus ise. Kuna tükid on piisavalt väikesemahulised, siis on lihtsam ka järge pidada rakendatavate reeglite üle.

Joonised

Robin Hunicke, Marc LeBlanc, Robert Zubek (2004) on öelnud, et head reeglid on sellised, kui neid on võimalik koheselt ka visualiseerida (Joonis 2). Tekitada joonised enne reegli rakendamist ning peale reegli rakendamist. Nii on näha, kas reegli rakendamisel on arendaja saavutanud soovitud tulemuse. Samuti lihtsustavad erinevad joonised, reeglitest arusaamist. Eriti kui on tegu keeruliste reeglitega. Jooniseid aga ei saa niisama lihtsalt luua. Tuleb tähelepanu pöörata sellele, kuidas reeglite rakendumine on kajastatud joonistel. Ehk kahe joonise vahel peab tekkima võrdlusmoment, kuidas

olukord on muutunud ning sellest tulenevalt peab arendaja läbi mõtlema, kuidas seda reeglit oleks võimalik rakendada reaalses mängus. Samuti peavad olema joonisel kujutatud objektid niimoodi paigutatud, et nende kontseptsioonist oleks võimalik aru saada. Mõtlema peab ka sellele, kuidas objekti kujutada mängijale. Näiteks võib objektile ümber joonistada mingi kuma, mis eristaks objekti ülejäänud keskkonnast.



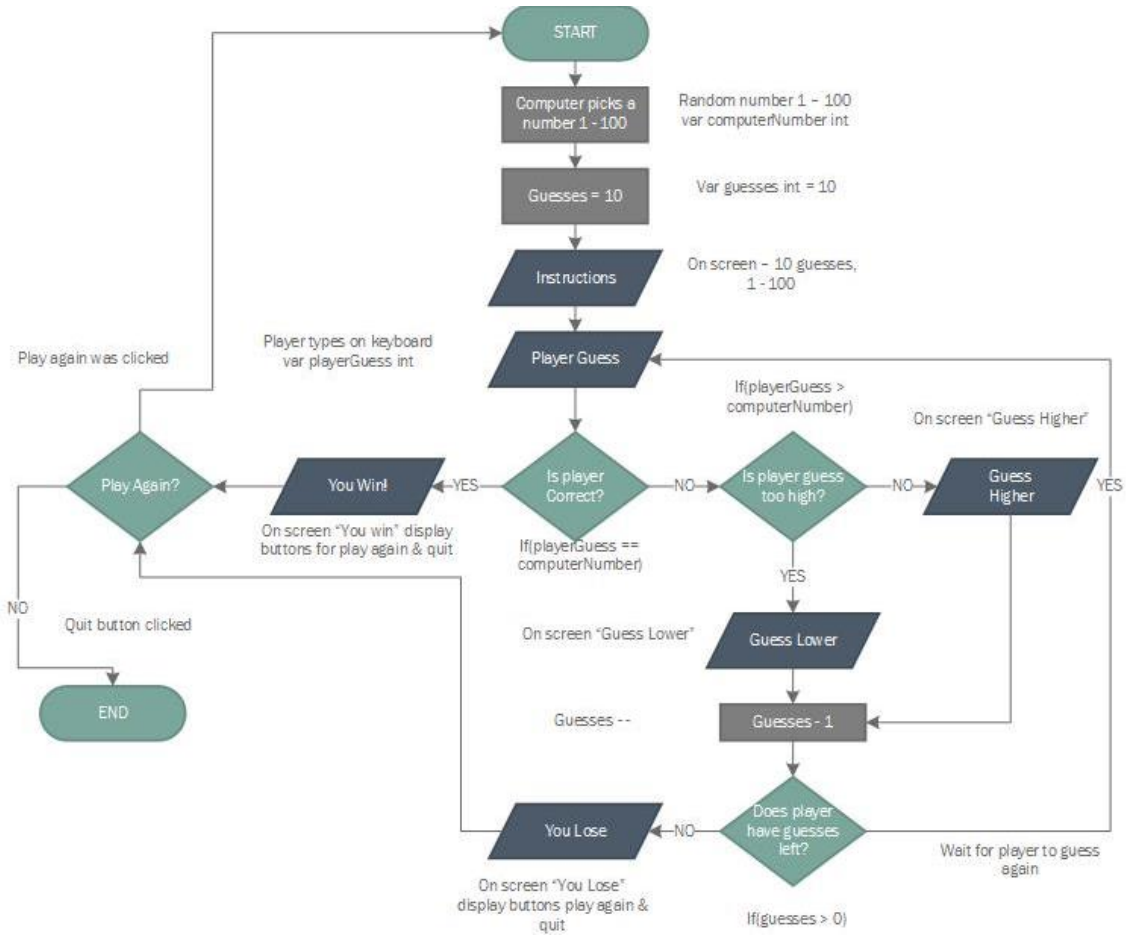
Joonis 2. Male nuppude liikumise reeglid

Joonisel on näha kuidas on kujutatud erinevate malenuppude liikumise võimalused. Igale nupule rakenduvad omad reeglid, kuidas see liikuda saab. Lihtne kujutlusviis, kuid selgelt edastab reeglid.

Vooskeemid

Kui reegleid rakendatakse objektidele nõnda, et mängijal on olemas valikuvõimalus, millist reeglit rakendatakse, siis on siinkohal hea rakendada vooskeeme. Ibrahim Yucel (2014) ütleb, et vooskeemidega (ingl *flow chart*) luuakse kindlad sammud selle kohta mis millalgi toimub ning millise interaktsiooni tulemusel. Skeemidel saab kuvada erinevaid edu ja ebaõnnestumise olekuid mille tulemusel rakenduks taaskord omad reeglid (Joonis 3). Vooskeemid samuti pakuvad parima ülevaate, kuidas liigutakse

erinevate olekute vahel. Keerulisemate reeglite kavandamisel asendavad vooskeemid paragrahvi formaadi.



Joonis 3. Numbri äraarvamismängu vooskeem

Joonisel on näha lihtsat numbri äraarvamismängu skeemi, kus arvuti genereerib ühe arvu kindlas vahemikus ning inimene peab selle arvu õigesti ära arvama. Kui inimene arvab numbri valesti, siis rakendub ebaõnnestumise rada ning õigesti arvamise korral rakendub õnnestumise rada.

Peab olema ka teadlik, millisele platvormile mäng luuakse. Erinevatel platvormidel on oma piirangud. Mobiilsed seadmed on kiiresti kasvamas oma võimsuses ja mängijaskonna arvus, kuid konsoolid ja arvutid on ikkagi veel tugevalt esindatud ning arvatavasti ei kao veel lähiajal kuskile. Igal platvormil on omad keerukused koodi kirjutamisel, piirangutel, interaktiivsusel ning isegi reeglitel, mida rakendada saaks (Klappenbach, 2015).

Juba eelnevalt sai mainitud, et oluline on ka see, milline saab olema mängužanr. Kas esimeses isikuvaates tulistamismäng, strateegiamäng või hoopiski midagi muud. Erinevatel žanritel on erinevad reeglid, mida saab rakendada ja ka erinevad tehnikad, mida kasutada saab.

2.2. Balansseerimise tehnikad

Reegleid peab mingitel viisidel hakkama balansseerima. Kõik mängus rakendatavad objektid peavad olema võrdsustatud ülejäänud loodava maailmaga. Tuleks vältida olukordi, kus mingid objektid on ilmselgelt tugevamad, kuigi neid oleks mängijal lihtne saada. On välja kujunenud kolm üldist tehnikat, kuidas balansseerida mänguobjekte.

Transitiivne suhe

Transitiivne suhe ehk kulukõver (ingl *cost curve*). See on kõige otsesem viis, kuidas balansseerida objekte. Üldine idee seisneb selles, et tuleb leida soovitud proportsioon asjade maksumusele, mis oleks suhtes objektide poolt saadavatest kasudest. See võib olla lineaarne proportsioon ehk objekt, mis maksab kaks korda rohkem on ka kaks korda kasulikum. Samuti võib suhe olla kumer mingil määral ehk mingil momendil rakenduvad lisatasud, et avada rohkemaid võimalusi, mida objekt mängijale pakkuda saab (Game Design Concepts, 2009). Need kumerused sõltuvad sellest millise mänguga tegemist on. Testimine annab aimu millisteks need kujunevad. Järgmiseks tuleks langetada kõik maksumused ning kasud numbrite tasemele, mida annab omavahel võrrelda. Võtta kokku kõik objekti maksumused ning liita need omavahel. Sama teha ka kasudega. Võrrelda neid kahte saadud arvu ning vaadata, kas objekt annab korrektse numbrilise väärtuse kasu poolest suhtes maksumusega (Game Design Concepts, 2010).

Seda tehnikat kasutatakse kõige enam kaardimängudes. Kui mängul on paigas kindel kulukõver, siis on palju lihtsam lisada mängu uusi kaarte. Näiteks mängus *Magic: The Gathering* on paigas kindlad reeglid ning maksumused, mida kaardid järgima peavad. Kui tahetakse luua uus koletise kaart, mille tugevus oleks 4 rünnakut ning 3 elu ja ta suudab lennata, siis on kindlad maksumused igal võimel, mis kaardil on. Lisades või eemaldades võimeid kaartidelt muutub ka nende maksumus, mis on märgitud kaardi ülemisel serval (Joonis 4).

Joonis 4. Magic: The Gathering kaartide võrdlus (autori joonis)

Joonisel on kujutatud kahte erinevat koletise kaarti. Esile on toodud, punaste ringidega, koletiste oskused. Igal oskusel on oma kindel maksumus, mis liidetakse kokku, ning seeläbi saadakse kaardi ülemises servas märgitud kogu maksumus. Kuna vasakpoolsel kaardil on ainult 1 rünnak ning 1 elu, siis see ei lisandu kaardi maksumusele juurde. Parempoolsel kaardil on küll ainult 1 rünnak, kuid 3 elu ning seetõttu, tuleb kaardimaksumusele 1 juurde. Samad arvutused kehtivad ülejäänud kaardil olevatele võimetele.

Intransitiivne suhe

Intransitiivne suhe ehk kivi-paber-käärid suhe. Selle suhte puhul ei pruugi olla otsest sidet maksumuse ning kasu vahel, vaid pigem on mänguobjektide endi vahel side (Gamasutra, 2007). Teisisõnu on olemas võimsamad mänguobjektid ning on nõrgemad mänguobjektid. Sõltub situatsioonist, kus objekti rakendatakse. Tuueski näiteks mängu kivi-paber-käärid, siis selles mängus ei ole ükski element dominante ning alistab kõik. Iga element on mingi teise elemendi suhtes tugevam, kuid kolmanda elemendi poolest jällegi nõrgem.

Seda tehnikat kasutatakse kõige enam strateegiamängudes. Mängijatel on üksused, mida nad loovad. Igal üksusel on omad maksumused, kuid neil on intransitiivne suhe teiste üksustega. Näiteks üks põhiline suhe üksuste vahel on selline, et vibumehed on tugevad

lendavate üksuste vastu, lendavad üksused on tugevad maaväe vastu ning maavägi on tugev vibumeeste vastu (Game Design Concepts, 2009). On näha selget kivi-paber-käärid lahendust. Sellest tulenevalt hakkabki mängima rolli juba mängijate endi oskused, kuidas nad paigutavad oma üksusi ning kuidas nad täidavad oma armee, et kõik rollid saaksid vastavas mahus täidetud.

Siinkohal saab isegi kombineerida transitiivseid ning intransitiivseid suhteid. Eelnevalt sai mainitud, et igal üksusel on maksumus. Kui näiteks lendaval üksusel on piisavalt kõrge maksumus, siis ta võib suuta alistada vibumeest, kellel võib madal maksumus olla. Sama klassi üksuste vahel on tegu transitiivse suhtega ning eri klassi üksuste vahel on tegu intransitiivse suhtega.

Puuviljane suhe

Omapärane nimetus balansseerimise tehnikale, kuid sellel on oma loogika, miks just selline nimetus. Põhimõtteliselt saab mänguobjekte ka selliselt luua, et need on nii erinevad üksteisest, seega otseseid võrdlusi on võimatu luua. Ehk tulles tagasi omapärase nimetuse juurde, siis põhimõtteliselt on võimatu ka võrrelda õunu ja apelsine (Game Design Concepts, 2009). Kuna formaalseid ning numbrilisi võrdlusi objektide vahel ei saa luua, siis ainuke lahendus leida balanss on läbi suuremahulise testimise. Näiteks võib asjade erinevus olla situatsiooniline, kus ühes olukorras on üks mänguobjekt teistest drastiliselt parem, kuid teises jällegi peaaegu kasutu.

Iga tehnikaga on seotud omad väljakutsed. Transitiivse suhte puhul peab arendaja leidma õige kulukõvera. Kui matemaatiliselt ei ole asi paigas, siis on arvatavasti ka kõik mänguobjektid balansist väljas. Kui üks objekt on balansseerimata, siis peab muutma kõiki objekte. Transitiivset suhet on palju lihtsam arendada peale testimist. Kuna palju sõltub matemaatilistest valemitest, siis sellega kaasneb palju katsetamisi, et asi õigeks saada - seega on see samuti ajamahukas viis (Game Design Concepts, 2009). Intransitiivse suhte puhul võib kivi-paber-käärid loogikaga üle pingutada, muutes kogu mängu ebahuvitavaks paljudele mängijatele. Paljude arvates on intransitiivsed suhted paljuski arvamismäng, mis kaasab endas juhuslikkust, ning seeläbi muutes mängu õnne põhiseks, mis ei vaja mängija endi oskusi (Gamasutra, 2007). Puuviljase suhte puhul on

tegu eriti keerulise tehnikaga, kuna ühte peamist balansseerimise tööriista ehk matemaatikat kasutada ei saa.

2.3. Suhtlus mängijate ja arendajate vahel

Mänguarendaja peab kindlasti olema ka kursis kindlate terminitega ning slängidega, mida mängijad foorumites või mujal kasutavad. Kui mängija pöördub mänguarendaja poole, siis suhtlus on enamasti vabas vormis ning seetõttu kasutatakse sõnu, millest võiks teadlikud olla. See haakub sellega, et on vaja palju testida eelnevalt mainitud tehnikates ning seetõttu on vaja aru saada testijaskonnast.

Kui asju testitakse kümnete ja isegi sadade inimeste poolt, võib ikkagi tekkida olukordi, kus mõni viga võib kahe silma vahele jääda. Kui mängijaid on kümneid või sadu tuhandeid, siis nii-öelda testijaskond kasvab märgatavalt. Kui inimesed edastavad oma kogemusi, siis on kasutusel mängurite seas kindlad slängisõnad, mida mõlemad osapooled võiksid teada, seega on oluline ka mänguarendajatel neid teada. Vaatleme üksikuid populaarsemaid slänge rollimängude žanrist.

Nerf

Termin millega vähendatakse mingi mänguobjekti efektiivsust või ihaldatavust. Tihti viidatakse selle terminiga taas rollimängudele. Kui lisatakse mängumaailma mingi objekt, mis võib algselt osutuda liiga tugevaks või liiga lihtsasti kättesaadavaks, muutes sellega mängu balanssi. Seetõttu tuleb muuta olukorda, kus mänguobjekt sobiks paremini mängumaailma (Laughead, 2016). Mängijatele ei pruugi see meeldida, kuid see on vajalik samm.

Buff

Vastand termin „nerfile“. Ehk kui mingi objekt on liiga nõrk või kergesti alistatav, siis tuleks seda natuke tugevamaks teha, et luua huvitavamat mängukogemust (Laughead, 2016).

Overpowered

Kui eelnevalt mainitud „buff“ terminiga on üle pingutatud ning mänguobjekt on liiga tugevaks muudetud, siis on tegu liiga tugeva objektiga, mille alistamine võib olla liiga keeruline või isegi võimatu. Tihti kasutatakse lihtsalt lühendit OP (Laughead, 2016).

Underpowered

Vastandtermin eelnevalt mainitud „overpowered“ terminile. Kui mingi objekt on liiga lihtsasti alistatav või mingi oskus on liiga nõrk, siis tuleks rakendada „buff“ terminit, et mänguobjekt uuesti tasakaalustada (Laughead, 2016).

Imba

Vastandsõna balanseeritud reeglitele. Kui mängija tajub, et kogetud olukord ei ole korralikult balansseeritud. Kas mingi objekt mängumaailmas on liiga võimas või mängu poolt juhitud vastased teevad liigselt sohki (Beepinoy, 2009).

Filler

Ebavajalik sisu mille eesmärk on ainult pikendada aega, mis kulub mängu lõpetamiseks. Loodud turundus eesmärkidel, et mängukarpide peale panna väide nagu mängupikkus oleks 50 tundi, millest tegelikult on ainult umbes 10 tundi väärt sisu (Laughead, 2016).

3. Balansseerimise protsess

Balansseerimise protessi käigus tuleb kasutada matemaatilisi valemeid ning arvutuskäike, et paika saada erinevate objektide omavaheline balansseeritud suhe. Selle suhte leidmine võib olla pikk ning vaevarikas teekond, kuid arendaja ise on see, kes peab kasutama instinkte ning taju, et määrata ära piirid, mil mäng on tema jaoks aktsepteeritaval tasemel tasakaalustatud (Burgun, 2016). Ehk millal arendaja ise tajub, et asjad tunduvad tema jaoks õiged. Küll aga peab kasutama teisi isikuid testimise protsessis, kes annavad omapoolseid soovitusi selleks, et balansseerimine oleks korrektselt läbi viidud. Arendaja töö on anda endast parim, et leida need tugevused ja piirangud, mida erinevad protsessid võimaldavad. Oluline on aga soovitus, et teha üks muudatus korraga (Game Design Concepts, 2009). Kui protsessi käigus selgub, et mingid elemendid vajaksid muudatusi, siis tuleks läheneda balansseerimisele aeglaselt. Viies korraga sisse hulgaliselt muudatusi, võib kogu mäng veel rohkem tasakaalust välja minna. Kui üks muudatus midagi katki teeb, siis on teada täpselt, milline muudatus kogu asja katki tegi. Kui viia korraga sisse kümme muudatust, siis ei ole arendaja kindel, milline muudatus tasakaalu segamini paiskas.

Oluline on kirja panna kõik loodud reeglid. Ka need, mis ei pruugi mängu jõuda. Nendest reeglitest võivad moonduda välja teised reeglid, mis jõuavad mängu. Reeglite loomine on paljuski katsetamine ning vigadest annab nii mõndagi õppida. Samuti kui kogu protsess on talletatud, siis tulevikus on lihtsam mänge luua, kuna arendajal on olemas juba mingi põhi millele toetuda ning eelnevalt tehtud vigade kordamine on välistatud (Game Design Concepts, 2009). Lõppude lõpuks säästab see aega ning frustratsiooni. Küll aga peab iga arendaja meeles pidama, et reeglite loomise ning balansseerimise protsess ei tohiks muutuda esmaseks prioriteediks mängu loomisel. Balansseerimise protsessid on tööriistad, mida arendaja saab kasutada, et mäng huvitavamaks muuta mängijate jaoks, kuid üldine mängust saadav lõbu ja rahulolu ikkagi oleneb ka teistest mängu aspektidest (Burgun, 2016). Erinevate testide läbiviimine on oluline, et välja selgitada, kas edasine tasakaalustamine võib hakata segama teisi mängudisaini elemente ning kahandama mängu kvaliteeditaset (Mandryka, 2015).

3.1. Paberprototüüpide testimine

Videomängude prototüübid edastavad mängu ideid ning mehaanikaid palju paremini kui erinevad kohtumised või disainidokumendid. Prototüübid säästavad aega ning raha. Vahel ei ole võimalik luua digitaalseid prototüüpe. Olgu nendeks põhjusteks tarkvaralised, riistvaralised, finantsilised või ajalised. Siinkohal saabki ära kasutada hõlpsasti kättesaadavaid kontoritarbeid (Marmura, 2008).

Kui arendajal puudub kogemus mingile platvormile arendamisel, siis hea koht, kuhu saab idee kirja panna ja läbi mängida, on paber. On mõttetu kulutada aega ja raha, et omandada oskused digitaalse prototüübi loomiseks, kui on olemas võimalus luua sama asi paberil. Samuti on võimalik paberprototüüpi esitada igas keskkonnas. Ei ole vaja luua keerulisi keskkondi, mida digitaalsed prototüübid tihti vajavad. Inimesed lähenevad paberprototüüpidele samuti palju meelsamini kui tehnoloogilistele lahendusele, seega saadav tagasiside saab olema samuti täpsem ning mahulisem. Ka paberprototüüpide loomisel on omad soovitused ja reeglid, et protsess oleks sujuvam.

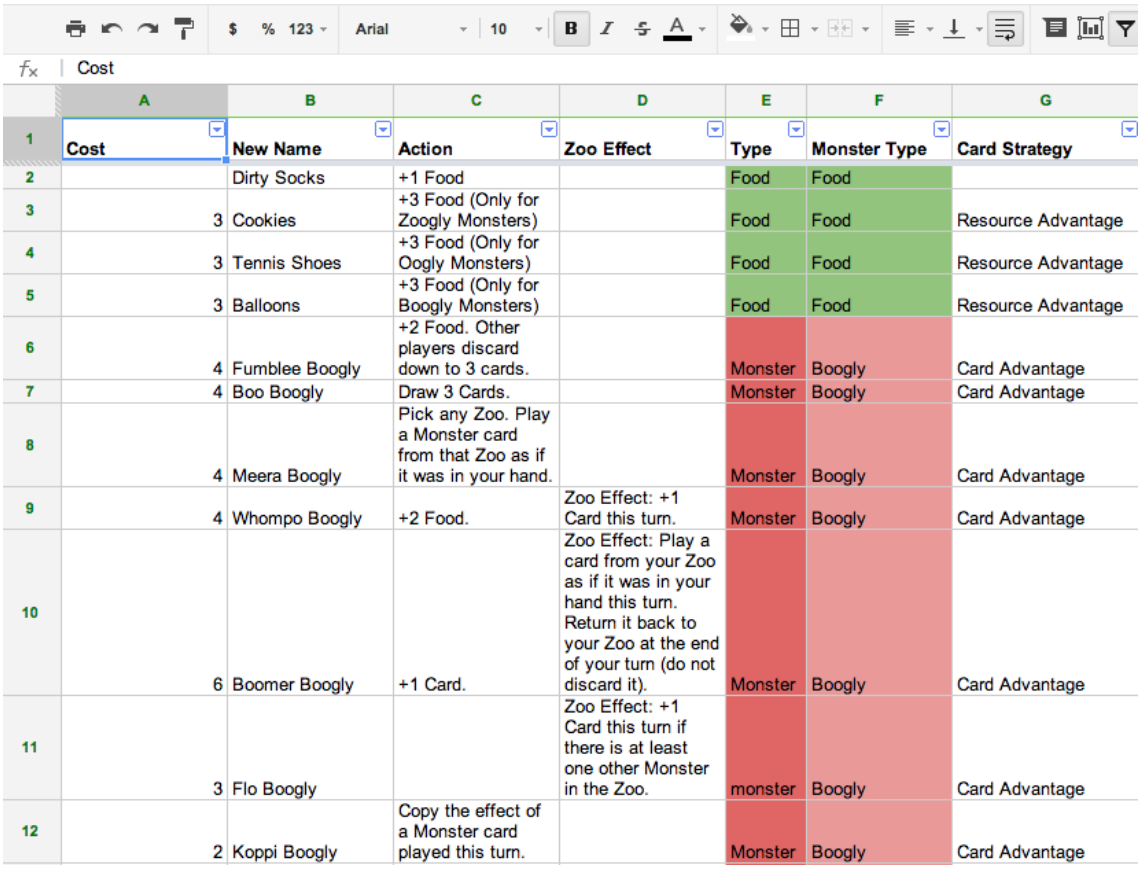
Nagu iga testi puhul on oluline paika panna standardsed protseduurid kindlustamaks, et isikud, kes testi läbi viivad ei mõjutaks kaudselt saadavaid tulemusi. Soovitavalt võiksid arendajad ise testi enne läbi mängida (Mifsud, 2012). Siis nad näevad, mida võiks testi protsessi veel lisada või eemaldada. Samuti saab aimu, kuidas oleks võimalik täita rolli, mida tehnoloogia tavaliselt täidab, ehk punktide arvetepidamine, võrguühendused jne. Kui on tegu väiksema arendajate meeskonnaga, siis testimise ajal on tavaliselt kõik isikud hõivatud mingi tegevusega. Seega on soovitatav jäädvustada testimine piltide ja videoga. See annab võimaluse hiljem tagasivaateid teha kogu protsessile (Marmura, 2008). Kui arendajad on rahul paberprototüübi ülesehitusega, siis saab testimise keskkonna avada suuremale hulgale testijatele.

Paberprototüübi selge eelis on selle kättesaadavus. Kõik tarbed, mida testides kasutatakse on enamasti odavad ning lihtsasti kättesaadavad. See annab võimaluse katsetada kõiki erinevaid aspekte mängust. Ka neid ideid, mis ilmtingimata ei jõua lõplikku mängu. Kui testitav komponent kukub testimisel läbi, siis on lihtne sellele kulutatud paber lihtsalt ära visata ning tutvustada uus komponent selle asemel. Kuna paberprototüüpide loomine ei vaja kindlaid oskusi, siis nende loomine toob kokku kogu

arendajate meeskonna. Igaüks saab panustada kogu protsessi, mis ühendab nii kogu disaini kui ka meeskonna.

3.2. Tabelites arvutuste testimine

Kuna matemaatilised valemid on balansseerimise protsessi juures väga olulised, siis on neid valemeid ka lihtsam rakendada tabelarvutusprogrammides. Ruby Cow Games (2013) nimeline ettevõte on jaganud oma kogemusi, kuidas nemad enda mänguobjekte haldavad tabelarvutus programmides (Joonis 5).



fx | Cost

	A	B	C	D	E	F	G
1	**Cost**	**New Name**	**Action**	**Zoo Effect**	**Type**	**Monster Type**	**Card Strategy**
2		Dirty Socks	+1 Food		Food	Food	
3	3	Cookies	+3 Food (Only for Zoogly Monsters)		Food	Food	Resource Advantage
4	3	Tennis Shoes	+3 Food (Only for Oogly Monsters)		Food	Food	Resource Advantage
5	3	Balloons	+3 Food (Only for Boogly Monsters)		Food	Food	Resource Advantage
6	4	Fumblee Boogly	+2 Food. Other players discard down to 3 cards.		Monster	Boogly	Card Advantage
7	4	Boo Boogly	Draw 3 Cards.		Monster	Boogly	Card Advantage
8	4	Meera Boogly	Pick any Zoo. Play a Monster card from that Zoo as if it was in your hand.		Monster	Boogly	Card Advantage
9	4	Whompo Boogly	+2 Food.	Zoo Effect: +1 Card this turn.	Monster	Boogly	Card Advantage
10	6	Boomer Boogly	+1 Card.	Zoo Effect: Play a card from your Zoo as if it was in your hand this turn. Return it back to your Zoo at the end of your turn (do not discard it).	Monster	Boogly	Card Advantage
11	3	Flo Boogly		Zoo Effect: +1 Card this turn if there is at least one other Monster in the Zoo.	monster	Boogly	Card Advantage
12	2	Koppi Boogly	Copy the effect of a Monster card played this turn.		Monster	Boogly	Card Advantage

Joonis 5. Mängu *Monster Zoo* kaardivalik Excel tabelarvutusprogrammis

Joonisel on näha, kuidas tabelarvutusprogrammi on lisatud mängus kasutatavaid kaarte. Igal kaardil on olemas nimetus, maksumus, oskused ning kaardi tüüp. See annab arendajale hea ülevaate mängus kasutatavatest elementidest. Samuti saab kaarte järjestada, erinevate kriteeriumite alusel, ning seeläbi näeb arendaja kuidas maksumused on erinevate elementide vahel jaotunud.

Näiteks rollimängudes on palju erinevaid relvi, rüüsid ja muid objekte, mida mängija võib mängumaailmast leida. Kui kanda need kõik ühte kohta, koos nendega seotud arvudega, siis on samuti lihtsam järge pidada. Rollimängude puhul on elemente väga palju seega on hea, kui need kõik ühes kohas ning sorteeritavad oleksid. Samuti saab teha koheseid arvutusi olemasolevate andmetega või neid vajaduse korral muuta.

Tabelarvutusprogrammides saab jooksutada ka statistilisi simulatsioone. Genereerides juhuslikke numbreid, saab luua erinevaid olukordi, kus objekte võidaks kasutada (Game Design Concepts, 2009). Näiteks kuidas genereeritakse mingi relva kahju tegemise numbrid ja sellest tulenevalt mingi rüü kaitse numbrid. Nende kahe objekti vahelise suhtluse läbimängimisel saame teada, kas rüü on piisavalt tugev rünnaku peatamiseks või relv on liiga tugev ja hävitab rüü. Kui simulatsioon laheneb arendajale meelepäraselt, siis on test edukalt läbi viidud ning objektid on omavahel balanseeritud.

Erinevate simulatsioonide läbimängimisel saab jälgida ka erinevate statiliste andmete muutumist ning sellest tulenevalt luua matemaatilised valemid. Kui võtta näiteks eelnevalt läbimängitud relva ja rüü simulatsioon, siis kas relva rünnak on ainult füüsiline või kaasneb sellega ka muid elemente. Samuti ka rüü puhul. Kas rüü kaitseb ainult füüsilise rünnaku puhul või on tal ka teisi boonuseid. Sellest tulenevalt saab muuta loodud valemeid ning seeläbi näha, kuidas ka teised väärtused muutuvad.

3.3. Tarkvara testimine

Üks moment peab loomulikult edasi liikuma igasugustest paberprototüüpide ning tabelarvutusprogrammide juurest ka reaalse mängu loomiseni digitaalsel kujul. Kõik eelnevalt läbimängitud olukorrad peab nüüd panema digitaalselt kirja ning hakkama neid sammhaaval rakendama mängus.

Oluline on rakendatavaid reegleid testida ka juba tarkvara tasemel. Võib arvata, et testimine viiakse läbi alles siis, kui mäng on valmis. Soovitatav on ikkagi testida erinevaid elemente juba siis, kui kood alles värskelt kirjutati. Taaskord rakendub soovitus, et viia muudatusi sisse ükshaaval, et oleks vähem eksimise ruumi (Sloper, 2016). Kui loodud koodi juures rakendatav reegel käitub eelnevates protsessides läbimängitud simulatsioonide kohaselt, siis võib edasi liikuda järgmise koodi osa

juurde. Kui aga ei käitu soovitud viisil, siis on lihtne koheselt muudatus sisse viia. Hiljem viia sisse suuremaid muudatusi võib kaasa tuua keerulisi olukordi, kus suurem kirjutatud kood võib ebamääraselt käituma hakata ning vigade otsimine võib muutuda väga pikaks protsessiks. See võib omakorda kaasa tuua suuremaid väljaminekuid ning väljalaske kuupäeva edasilükkamise. Väikeste arendajate jaoks võib see muutuda suureks tagasilöögiks.

Algses faasis on testijaskond seega pisike. Ehk arendajad ise testivad pidevalt oma loodud koodi. Hiljem, kui mäng saavutab *alpha* taseme ehk mängu on võimalik juba mängida, kuid paljud loodud elemendid on puudulikud, katki või umbmääraselt rakendatud, siis võib kaasata testimise protsessi juba rohkem inimesi. Selleks tuleks luua testiplaan, mida iga inimene jälgib (Sloper, 2016). Testiplaanis võiks kirjas olla, mida arendajad soovivad testida ning ka elemendid, mida arendajad teavad, et on katki. Hea testiplaan tõstab arendajate efektiivsust, et vigaseid elemente parandada. Kui mäng on saavutanud juba *beta* taseme, kus mängu on rakendatud veel enam mängitavaid elemente ning on parandatud eelnevalt teavitatud vigu, siis testijaskond kasvab veelgi (Hobarg, 2014). Uued testijad, kellel puudub varasem kogemus loodava mänguga, võivad leida uusi vigu, mida eelnev testijaskond ei pruukinud leida. Testijaskond samuti määrab lõpuks, mis elemendid jõuavad või ei jõua lõplikku mängu. Kuna testijaskond võib olla siinkohal juba üpris suur, siis on soovitatav määrata testi läbiviivad isikud, kes koordineerivad testijaid.

Tagasiside peab olema korrektselt dokumenteeritud, et arendajad saaksid oma keskkonnas taasluua olukordi, kus vead tekkisid. Ebatäpselt esitatud dokumentatsioon võib tähendada, et mingid vead lõpuks jõuavad ikka lõplikku mängu. Arendajad hindavad ka vigade kriitilisust (Sloper, 2016). A astme vead on kriitilise tähtsusega. Need on vead, mis võivad väljalaske kuupäeva edasi lükata. Vead, mis võivad mängija arvuti tarkvara katki teha. Selliseid olukordi tuleb ilmtingimata vältida. B astme vead on olulise tähtsusega, mis vajavad tähelepanu, kuid mäng on ikkagi mängitav. Mitu B astme viga on võrdväärsed A astme veaga. C astme vead on väikesed ning ebaselged vead, mis on pigem soovituslikud parandused kui vead.

Kui mäng lõpuks tehakse kättesaadavaks kõigile lõpliku mängu kujul, siis testimise protsess ei ole tegelikult lõppenud. Tuleb olla valmis uueks ning ka palju suuremaks

tagasisideks. Kuna vead võivad ikkagi jõuda ka lõplikku mängu, siis arendajad peavad valmis olema parandama ka neid vigu (Sloper, 2016). Nädalad või isegi kuud peale väljalaset tuleks need vead ära parandada. Isegi kui on möödunud pikem aeg mängu väljalaskest ning edasi on liigutud uute mängude juurde, siis tuleb valmis olla, et tagasi minna eelneva mängu juurde, kui avastatakse mingi uus ootamatu viga.

Kokkuvõte

Mängureeglite rakendamine on äärmiselt oluline mängudisaini osa. See ühendab erinevaid elemente ning kui need ei ole rakendatud korrektselt, siis võib kogu ülejäänud mängudisain kokku variseda. Mängu arendades tuleb endale selgeks teha, milline saab loodava mängu žanr olema. Mängužanr määrab ära, milliseid reegleid peab arendama hakkama. Eriti hea oleks siinkohal, kui on teada ka, milline saab olema mängumaailm. See omakorda lihtsustab rakendatavate mängureeglite elementide valikut. Kui need algsed elemendid on paigas, siis võib mängu looma hakata. Kui mängu loomise protsess on sealmaal, kus mängu saab juba algtasemel mängida, siis võib hakata mõtlema elementide taga olevatele numbritele. Kui tugevad mingid elemendid on ning kui keeruline mingi ülesande lahendamine olla võib. Ehk teisisõnu kuidas neid elemente balansseerida. Töö kirjutamisel kujunesid välja minupoolsed soovitused tulevastele mänguarendajatele:

1. Valida, milline saab mäng olema. Kas *paidea* või *ludus* tüüpi. See kitsendab mängužanri valikut ning ka millistele reeglitele tuleks keskenduda.
2. Milline saab olema mängužanr? Erinevatel žanritel on erinevad reeglid, mida rakendada.
3. Leida endale meelepärane viis, millist tehnikat kasutada reeglite balansseerimisel. Kas kulukõver, kivi-paber-käärid või puuviljane? Milline on sinu mängule sobivam.
4. Testi, testi ja veelkord testi. Iga loodava elemendi valmimisel testida, kuidas see suhtleb olemasolevate elementidega, kuidas see sobitub olemasolevasse maailma ning kui keeruline oleks seda alistada. Ei pea kohe rakendama koodi, vaid kasutama kättesaadavamaid ning lihtsamaid ressursse testide läbiviimiseks. Näiteks teha paberprototüüpe või kanda objektid tabelarvutusprogrammi.
5. Elementide taga olevad numbrid koodis peavad olema balansseeritud. Kui kaks erinevat elementi üksteisega vastamisi lähevad, siis kui tugevad nad teineteise suhtes on.
6. Kui mäng võib olla arendaja arvates valmis, siis tuleb olla valmis ootamatusteks, kui mäng jõuab mängijate kätte. Välja võivad tulla uued vead või muud probleemid, mis tuleks parandada.

Teema valimisel algselt arvasin, et mängureeglite balansseerimine on enamasti ainult numbrite korrigeerimine, kuid töö kirjutamisel selgus, et see ei ole nii. Iga elemendi välja valimine mängureeglite loomisel ning rakendamisel on osa suuremast balansseerimise protsessist. Mänguarendajatele on see hea ülevaade sellest, mida tuleks jälgida enda loodavas mängus selles osas, mis puudutab mängureegleid.

Töö annab ka aimu, et mängu loomise protsess algsest mõttest kuni valmistooteni on tegelikkuses keerulisem. Kui juba ainuüksi reeglite rakendamisel ning balansseerimisel on nii palju elemente, mis vajavad pidevat tähelepanu, siis eeldan, et ka teised protsessi osad vajavad samasugust tähelepanu.

Kuna tarkvaraarenduses ei ole kindlaid fakte selle kohta, mida ei või ja mida võib teha, vaid on olemas parimad praktikad, siis ka minu töö tugines nendele parimatele praktikatele. Seda võib ignoreerida, kuid neid soovitusi järgides saavutatakse parem tulemus.

Kui tehtud tööga suudan kasvõi ühe tulevase mänguarendaja rohkem mõtlema panna rakendatavale reeglitele, siis on olnud töö edukas.

Kasutatud kirjandus

Ang, C.S. (2005). *Rules, Gameplay and Narratives in Video Games*. Loetud aadressil https://www.researchgate.net/publication/236870305_Rules_gameplay_and_narratives_in_video_games

Beepinoy. (2009). *What IMBA Means in Online Gaming World*. Loetud aadressil http://www.beepinoy.com/2009/12/30/what-imba-meaning-in-online-gaming-world/

Burgun, K. (2016). *Understanding Balance in Video Games*. Loetud aadressil http://www.gamasutra.com/view/feature/134768/understanding_balance_in_video_.php?print=1

Chen, J. (2006). *Flow in Game* (magistritöö). Loetud aadressil http://www.jenovachen.com/flowingames/Flow_in_games_final.pdf

Frasca, G. (2003). *Simulation versus Narrative: Introduction to Ludology*. Loetud aadressil http://www.ludology.org/articles/VGT_final.pdf

Gamasutra. (2007). *Rock Paper Scissors - A Method for Competitive Game Play Design*. Loetud aadressil http://www.gamasutra.com/view/feature/130150/rock_paper_scissors__a_method_for_.php

Game Design Concepts. (2009). *Level 16: Game Balance*. Loetud aadressil https://gamedesignconcepts.wordpress.com/2009/08/20/level-16-game-balance/

Game Design Concepts. (2010). *Level 3: Transitive Mechanics and Cost Curves*. Loetud aadressil https://gamebalanceconcepts.wordpress.com/2010/07/21/level-3-transitive-mechanics-and-cost-curves/

Gingold, C. (2003). *Miniature Gardens & Magic Crayons: Games, Spaces, & Worlds* (magistritöö). Loetud aadressil http://levitylab.com/cog/writing/Games-Spaces-Worlds.pdf

Hobarg, J. (2014). *Differences between Software Testing and Game Testing*. Loetud aadressil http://www.gamasutra.com/blogs/JohanHoberg/20140721/221444/Differences_between_Software_Testing_and_Game_Testing.php

Hobby Gamedev. (2011). *Videogames and rules*. Loetud aadressil http://www.hobbygamedev.com/adv/videogames-and-rules-part-2/

Hunicke, R., LeBlanc, M., & Zubek, R. (2004). *MDA: A Formal Approach to Game Design and Game Research*. Loetud aadressil http://www.cs.northwestern.edu/~hunicke/MDA.pdf

Juul, J. (2001). *A Clash between Game and Narrative* (magistritöö). Loetud aadressil http://www.jesperjuul.net/thesis/AClashBetweenGameAndNarrative.pdf

Klappenbach, M. (2015). *Choosing The Video Game Platform Best For You*. Loetud aadressil https://www.lifewire.com/choosing-video-game-platform-811337

Laughead, L. (2016). *Video Game Vocabulary, Jargon, and Slang*. Loetud aadressil http://www.leelaughead.com/video-game-vocabulary-jargon-and-slang/

Lidwell, W., Holden, K., & Butler, J, (2010). *Universal Principles of Design: 125 Ways to Enhance Usability, Influence Perception, Increase Appeal, Make Better Design Decisions, and Teach through Design. 2nd Ed*. Minneapolis: Rockport Publishers

Make-Video-Games.com. (kuupäev puudub). *Lesson 1 Continued: What are video games? - Objects in games*. Loetud aadressil http://www.make-video-games.com/game-school/1b-introduction-objects-in-video-games.htm

Mandryka, A. (2015). *Tips on game balancing*. Loetud aadressil http://gamewhispering.com/tips-game-balancing/

Marmura, R. (2008). *Paper Prototyping: 5 Facts for Designing in Low-Tech*. Loetud aadressil http://gamecareerguide.com/features/622/paper_prototyping_5_facts_for_.php

Mifsud, J. (2012). *Paper Prototyping As A Usability Testing Technique*. Loetud aadressil http://usabilitygeek.com/paper-prototyping-as-a-usability-testing-technique/

Ruby Cow Games. (2013, 12. veebruar). *Excel and Google Docs Spreadsheet Tips for Game Designers* [ajaveebipostitus]. Loetud aadressil http://rubycowgames.com/excel-and-google-docs-spreadsheet-tips-for-game-designers/

Sloper, T. (2016). *Tester – The Unsung Heroes of Games*. Loetud aadressil http://www.sloperama.com/advice/lesson5.html

Tosca, S. (2003). *The Quest Problem in Computer Games*. Loetud aadressil http://www.it-c.dk/people/tosca/quest.htm

Tvtropes. (kuupäev puudub). *The Computer Is a Cheating Bastard*. Loetud aadressil http://tvtropes.org/pmwiki/pmwiki.php/Main/TheComputerIsACheatingBastard

wikiHow. (kuupäev puudub). *Designing Gameplay*. Loetud aadressil http://www.wikihow.com/Design-a-Video-Game

Yucel, I. (2014). *Rules For Writing Rules: How Instructional Design Impacts Good Game Design*. Loetud aadressil http://analoggamestudies.org/2014/10/the-rules-for-writing-rules-how-instructional-design-impacts-good-game-design/