DESIGNING GAME ANALYTICS FOR A CITY-BUILDER GAME

Karoliina Korppoo

University of Tampere
School of Information Sciences
(SIS)
Information Studies and
Interactive Media
Master's Thesis
May 2015

UNIVERSITY OF TAMPERE, School of Information Sciences (SIS)
Information Studies and Interactive Media
KORPPOO, KAROLIINA: Designing Game Analytics for a City-Builder Game
Master's thesis, 74 pages
May 2015

The video game industry continues to grow. Competition is tough as games become more and more popular and easier for the users to get, thanks to digital distribution and social media platforms that support games. Thanks to the readily available internet connections and games using them, data of player behaviour can be acquired. This is where game analytics come in. What sort of player actions provide meaningful information that can be used to iterate the game? Typically game analytics is applied to multiplayer games or free-to-play social network games, not to single player games.

This case study focuses on the design of game analytics for *Cities: Skylines* (Paradox, 2015), a modern city-building sandbox game. The writer of the thesis is the lead designer of *Cities: Skylines*, so the perspective to the work is from inside the games industry and on how metrics are designed alongside game design.

The results of this study show that some practices used with free-to-play game's analytics can be used with more classic games, but that sandbox simulation games are very challenging for game analysis. This is due to how sandbox games strive to support player self-expression and cater to many different playing styles.

Keywords: Video Games, Games, Game Analytics, Metrics, Telemetry, Game Industry, Cities: Skylines

Table of Contents

1 INTRODUCTION

The video game industry continues to grow. Competition is tough as games become more and more popular and easier for the users to get, thanks to digital distribution and social media platforms that support games. With internet connections becoming ubiquitous, most games utilize them by offering automatic updates, multiplayer features or other online features, even for single player games. Thanks to the readily available internet connections and games using them, data of player behaviour can be acquired. This is where game analytics come in. What sort of player actions provide meaningful information that can be used to iterate the game? Typically game analytics is applied to multiplayer games or free-to-play social network games, not to single player games.

This case study focuses on the design of game analytics for *Cities: Skylines* (Paradox, 2015), a modern city-building sandbox game. The writer of the thesis is also the lead designer of *Cities: Skylines*. The Lead Designer in a project is responsible for designing the game system: how the game behaves, how user interactions affect it, what can exist in the world and what cannot. The Lead Designer works together with the Lead Artist and Lead Programmer to make sure all pieces of the game fit together. The main task for the Lead Designer is to make sure the game is interactive and logical, the world is coherent and player actions have consequences. The work mostly includes writing texts and tables to describe the game system and the components in it and their attributes. Due to the writer's position, the thesis looks at designing game analytics from within the games industry and from the perspective of a person working for a company to create the metrics system for game analysis. As the Lead Designer's job is not only to handle metrics but to design the game system, this occasionally overlaps with metrics and metrics are not the first priority for the company. The thesis describes work done by the developer when designing telemetry, the background of people involved, the reasons for including metrics collection and the aim of the data. Analysing actual metrics gathered from the game reveals information of player behaviour, daily active users and the number of crashes.

The main questions to be answered are 1) how much of the literature concerning game analytics can be applied to a game that does not utilize the free-to-play revenue model, and 2) are sandbox simulation games special in using analytics and if yes, how. Sandbox games are very different from social network games and generally appeal to wide audiences, so applying game analytics to them would help to gain useful information about the sandbox genre and the player’s playing sandbox games. If game analysis literature concerning the free-to-play revenue model can be applied to sandbox games, it is most likely usable also for other genres that utilize different revenue models.

2 GLOSSARY

This section explains some of the main terms used in the text.

Developer - A video game developer is a software developer that specializes in video game development (Bethke, 2003)

Free-to-play game - A game that has no initial cost, user can acquire it free of charge and play the core game for free. User is offered virtual goods to purchase, such as energy boosts and virtual clothes. Selling goods is the way free-to-play games make revenue. (Paavilainen et al., 2013)

Game analytics - A process of discovering and communicating patterns in data with the express purpose of solving problems. (Drachen, 2013)

Game metric - A quantitative measure of something related to games. For example, a measure of how many daily active users a social online game has; a measure of how many units a game has sold last week; a measure of the number of employee complaints the past year; task completion rates in a production team for a specific title, etc. – these are all game metrics because they relate directly to some aspect of one or more games. (Drachen, 2012)

Micropayment - A very small payment, mostly used when trading virtual goods. Amount may vary according to context. (Paavilainen et al., 2013)

Monetization - An umbrella term for different business practices for gaining profit with a game product (Drachen, 2013)

Publisher - A company that publishes video games that they have either developed internally or have had developed by a video game developer (Wikipedia)

Telemetry - The term we use for any source of data obtained over a distance, which is used in game development. There are many such sources, with some of the most popular being user telemetry, i.e. data collected from installed clients or servers about the behaviour of users. For example, what items they purchase, how much they play and when, interactions with other users, etc. (Drachen, 2012)

3 GAME ANALYTICS

Game analytics mean the process of collecting and analysing information via telemetry. Telemetry in games is the process of reading information about players' actions from another location. Usually it is done by collecting metrics and sending them to a server, from where an analyst can look at the data, compare different metrics and to try to find out if things are working as intended and if not, why. Metrics are not to be confused with automatic bug reporting, which also sends data from the player's machine to servers in case of something going wrong. Metrics are statistics of gameplay, concentrating on what the player is doing, how much and when. Bug reporting is more about technical problems than player behaviour and aims at sending information regarding the cause of the bug.

When working on a boxed single player game, analytics can be used for several things. For a publisher, gathering and analysing game metrics is important in choosing future products or additions to the current product. It is much easier to decide on what sort of content would interest the existing players, when you can see data of what things people do most and least in the game.

For a developer, seeing if the game is played as intended helps with tracking bugs, balancing and confirming design choices. If the product is sold without further support, the only thing a developer can do is to learn from the metrics for future projects.

Using analytics is very different for self-published social online games that are the main area game analysis is used in. Monetization in social games relies on good pacing, and pacing can best be measured with metrics collection and analysis. In this paper the metrics collection is planned for a single player, offline, boxed game published with the traditional revenue model (players buy a finished product and are not required to make any more payments), so optimizing the game for monetization is not of importance. The game being single player means that there is no need to worry about balancing out skill differences inside the game, offline play makes it necessary to collect data packs to send to the server instead of a continuous data stream, and a boxed game stands for a product which the user buys as a whole, finished product and expects to play as is. Most of the texts concerning game analytics are focused on social game design, which cannot really be applied to this project because of the totally different monetization model, very different play sessions and because *Skylines* is not focusing on any social content or acquiring new users via game features.

The basic concept of game analysis is that data is collected and looked at analytically. One definition is by Drachen et al. (2013) “Analytics is the process of discovering and communicating patterns in data, towards solving problems in business or conversely predictions for supporting enterprise decision management, driving action and/or improving performance”. Davenport & Harris (2007) define analytics to mean not just querying and reporting, but using actual scientific methods to make sense of the data and obtain valid information. Drachen et al. also point out that the purpose of game analytics, and analytics in business in general, is to gain business intelligence and allow businesses to become data-driven in choosing practices and strategies. They recognize many different areas of data for business intelligence in the information and communications technology field, of which the game industry is a part of. These areas include the market, the company itself and the users. Game analytics combines data from these areas and focuses on using it for game development and game research.

Bilas (2014) divides game analytics into subcategories in her talk on how anyone can analyse data. These categories describe what type of analytics can be applied to data or done using data.

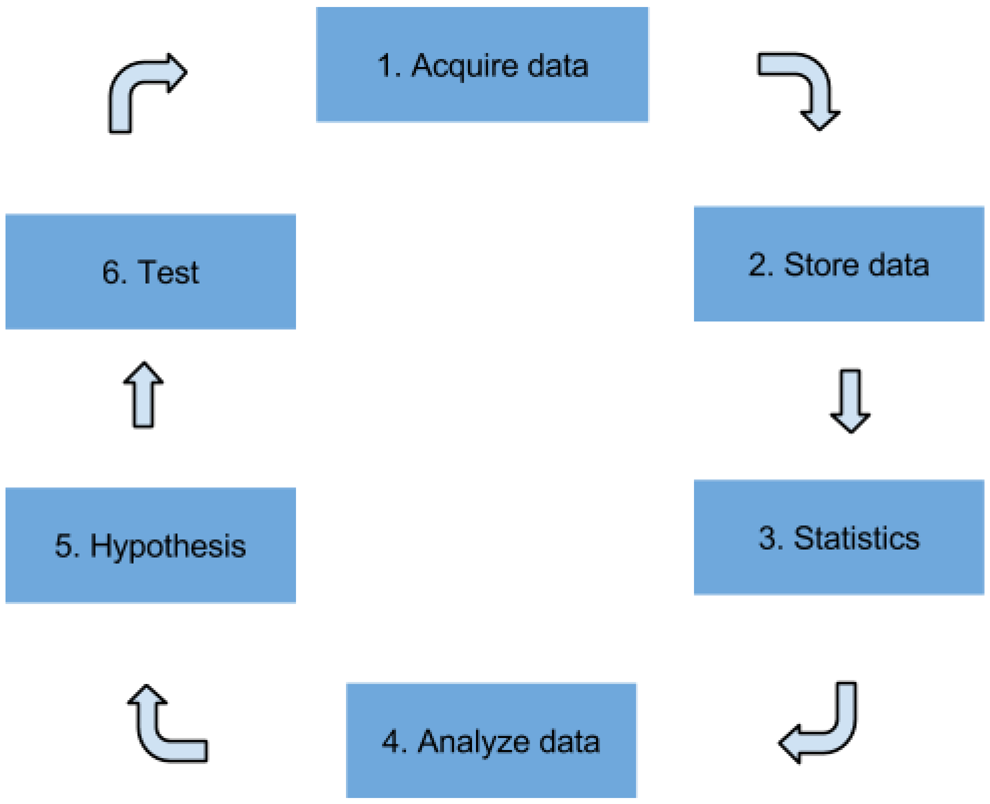
Table 1. Types of analysis, original table by Bilas (2014)

	Observe				Experiment
	Easy			Hard	
Type of analysis	Descriptive	Exploratory	Inferential	Predictive	Causal
Definition	Quantitatively describing data	Looking for previous unknown relationships in data	Testing theories with a sample of data	Analysing current events to predict future events	Measuring what happens to one variable when you change another
Method	Distribution; 5-number summary; Before/after;	Directionality; Visualizations	Regression models; Chi squared	Modeling, machine learning, data mining	A/B testing

Bilas’ categories (table 1) are further divided into analysis types that observe data and analysis that needs experimenting. Descriptive, exploratory, inferential and predictive analyses’ fall under the observing category, because all of them need a mass of data and results are gained by observing the data by using different methods in these categories. None of these analyses’ need touching the actual product or creating additional content, just gaining data and using a method to find out what the data means. Causal analysis in under the experiment category, because it differs from the other methods by needing content and requiring more data to be gained to complete the analysis.

Bilas goes on to mark the categories easy or hard based on how much expertise executing the analysis successfully is needed. All others are easy, except predictive and causal analysis. Bilas describes predictive analysis as being technical and data heavy, and to concentrating as the name says on things that the users might do in the future based on things that they are currently doing. Bilas concentrates on the easy analysis and a thing that all or most companies can do, meaning that the hard ones require resources that most companies do not have. For *Cities: Skylines*, the only analysis done by the developer is descriptive, simply because there is no time allocated for further analysis and only very few resources available.

Kennerly (2003) describes game analytics as an ongoing process with six repeating steps. While Kennerly's article is quite old, the basics still seem to apply. Kennerly’s visualization of game analysis as an iteration process (picture 1) clearly shows how using analytics works best for an iterative process and is part of a system of improving the game cycle by cycle.



Picture 1. Kennerly's take on game analysis as process of recycling old data into new design. Adapted from Kennerly (2003).

3.1 Benefits of Game Analytics in Design

Hazan (2013) writes in his article *Contextualizing Data* on how a shooter game *Crackdown* (Microsoft, 2007), went through game test sessions and user questionnaires to confirm design choices and make the game more fun. This example was exceptionally interesting, since the fun factor is something that is very hard to measure in games. Because of this, designers tend to rely on their own knowledge and experiences to design game features that are fun. The whole concept of fun is hard to define and fun is also a very personal experience; what one user thinks is fun might not be fun for another user.

With *Crackdown*, players were asked to rate how fun the game was overall and answer an open question about what was most fun in the game. *Crackdown* is an open-world role-playing game where users take the roles of super-cops who fight crime. When the playtest data and questionnaires were analysed, many players specifically mentioned that they found jumping very high to be fun. When the fun ratings were compared with player characters' abilities, it became apparent that players whose characters had high agility scores, allowing them to jump higher, were the very same users who usually gave the game a high fun rating. Users who did not get to enjoy high jumps because their characters had low agility skills, tended to rate the fun factor of the game lower.

One of the key features in *Crackdown* was the ability to be a super-cop with skills surpassing those of normal humans. According to the developer, it was a design intention to provide a super-cop feeling in the game. The thought that super human powers were fun was thus confirmed by game analytics. Based on the analysis, the game was changed so that gaining a high agility skill was easier than before and players were encouraged to do so more than before. Another round of playtests was then done, and the average fun ratings went up as expected.

The *Crackdown* example is a great way to show how analytics can help confirm design choices and to identify factors which are important to users. As for business intelligence in general, designers benefit greatly from almost any information concerning the audience and previous games of the same genre.

From personal experience as a game developer, I can say that many game projects start very fast and have very limited time for background research. This means that developers use their intuition in deciding what works for the game and what does not. Actual data of user behaviour in a similar title would be a tremendous help when starting out with a new game project. Choosing what features to develop would still need experience and expertise, as low activity levels in some features could mean that the feature is not interesting to users at all in the genre or that it has been executed poorly. A good tactic might be to choose one or two popular features and two or three unpopular features that seem like things that could be spiced up and done differently in order to reach their full potential. Competing with a highly successful title by doing all the same features in the same way would not be wise unless you have access to a larger budget and/or more resources than they had. This is naturally highly unlikely if you work in a small studio.

In the absence of data, the usage and popularity of different features have to be defined by designers in the following ways: guessing based on own experience, playtests (results analysed based on own experience) or analysing forums, reviews and scores, again based on personal experience. This places a big strain on the guesses and choices made by the designer and is likely one the reasons designers have traditionally first worked for years in other positions in the industry, familiarizing themselves with games well enough to make educated guesses. An experienced team with lots of knowledge between them is a great way to peer review design ideas, but there is still a lot of responsibility on the designer to make right enough choices at the start of the project so that precious work hours are not wasted on designs that get scrapped or significantly altered later.

Drachen et al. (2013) discuss examples of metrics for different game genres. The number of example metrics is not the same for all game types, with the first person shooters having the most suggestions (18 metrics) and simulation games the least (1 metric). This can be interpreted to reflect the differences between the genres. Shooter games have very similar mechanics and not many options on how to solve problems. The things to measure are mainly related to what sort of weapons work best, how long completing levels takes for the player and where they are situated on the maps. Simulation games are a very wide genre, ranging from realistically modelling how to fly an airplane to cartoonish depictions of running a farm.

This does not mean metrics are not applicable or do not benefit simulation games. Simulation games generally have a long life compared to games with a focus on narratives, as the games do not have a clear ending point and the player can explore the simulation within set rules. For a city-builder simulation game, such as *Cities: Skylines*, player actions are more varied than in shooter games. There are many ways for the user to solve problems, and the act of choosing what to do and seeing how the results of the choice persist in the world are a big part of the whole game experience. Also the player has no clear avatar, there is not the same kind of moving through maps or even a level structure. The player is simply given a sandbox with tools that they can play with. Playing includes building, looking at the results and finding problem areas, fixing the problems and choosing which things to fix next. The city the player is building always tries to stay a little bit imbalanced, never perfect, so there is always a pressing matter for the player to handle. Sometimes, if the user wants to build something that costs a lot of game currency, playing consists of first adjusting the city to be in a state that produces money, and then simply waiting for the sum to accumulate. Playing is slow-paced and the game can be paused if the user wishes to ponder on the best course of action. Slow paced, long playing sessions that contain various ways to solve a problem make for a lot of metrics. Choosing which metrics are the ones to look at is what this paper is all about.

3.2 Related Work

The amount of material written on game metrics is fairly small. Much of the information is held by companies and used mostly internally. This is due to metrics information not being very useful to players and the possible use of the data by competitors. For social games, the most valuable part of the game is the system with which players are kept interested in the game so they keep coming back, so publishing the data about these polished core features would only help the competition in making their products better. There is a middle ground, however. For example, Blizzard has an open database of all *Diablo 3* (Blizzard Entertainment, 2012) characters. This data is very useful to players when presented clearly. One site to do so is *Diablo Somepage*, where lists of the most popular gear and skill choices, divided into character classes, are compiled. All of the data is collected from the game publisher's open database. These are very simple metrics, not really trying to look into player actions or motivation, but just list what items and skills are useful in the game. This brings the game some new depth, as players can plan playing even when not logged in to the actual game.

Technical skills for collecting metrics are not rare, basic knowledge of programming servers is enough to get data saved. Making the data readable and easy to compare the metrics with other metrics requires more skill. Game analytics is a new and valuable skill, and has not yet received much academic attention. Related topics are in the realm of design research, which can be applied to the work done on *Skylines* easily.

So far the most comprehensive book on game analytics is *Game Analytics - Maximizing the Value of Play Data* (2013) by A. Drachen, M. Seif El-Nasr and A. Canossa. The book has articles, case studies and interview that offer both a good overview of the field and very specialized information on some subjects.

An online resource on the subject is the *GameAnalytics Blog* run by the company called GameAnalytics, which offers solutions for companies looking into integrating telemetry and analysing data. Drachen, who also has a large role in the book *Game Analytics - Maximizing the Value of Player Data* (2013) is the lead game analyst of GameAnalytics and also publishes articles on the blog.

Because of the fast pace of the games business, much of the information on game analytics is available as presentations given at events (filmed and uploaded to YouTube or some other video sharing platform) or articles on industry magazines or websites. Academic research done on actual cases of analytics used are still hard to find. Some case studies are presented in *Game Analytics*, but they are mostly based on talks given at industry events, not academic studies. Analytics seems like it is the up and coming trend to take on the game industry in the wake of free-to-play games and social network games success stories that raise much interest for metric driven design and the possibilities it offers. Jeferson Valadares, the then General Manager of Games at *Flurry* analytics app tool, previously Studio Director of *Playfish* stated in a 2011 talk that games companies should "measure or die". *Playfish* created very successful social games, such as *The Sims Social* (2011) with over 65 million players (Wikipedia, original reference no longer available). Valadares (2011) feels that analytics are the next big step in the continuum of mobile games. He places it on a time line with historical events, like colour screens and the coming of the iPhone. While Valadares talks mainly about mobile games and social games, some points can be expanded to apply to games as a whole. His notion is that analytics is something not only games use, but that games as entertainment products are in an unique position to be able to constantly improve based on user feedback and data gained. Once a song, for example, is released, it is done and is highly unlikely to change based on user feedback. Games can be changed, and with digital distribution it is fast and easy for PC games as well as mobile applications. The *Playfish* method of creating social games clearly works, as shown by the high value of the company and their player numbers. It should be noted, however, that the *Playfish* studio no longer exists: *Electronic Arts* acquired it in 2009, and closed the studio on 2013 and retired all *Playfish* created games. *Playfish* games do not live on even when they did measure, but the practice of measurement, analytics and metrics driven game design is still going strong.

4 METHOD

This paper takes a look at how metrics were designed for *Cities: Skylines,* developed by *Colossal Order* and published by *Paradox Interactive* in March 2015. *Cities: Skylines* is an offline single player city-builder game. Original notes on metrics chosen by the developer in one marathon meeting are listed and explained in this chapter, metric by metric. The reason why all metrics were decided upon in one meeting is simply time pressure. The need to implement metrics was encountered fairly late in the production, and working on metrics that were not required by the publisher was deemed to be something that can only take up a minimal amount of work time. The schedule allowed for one day to be used on additional metrics, so the development team members interested in them prepared ideas on their free time and then gathered for one long meeting to settle which ones would make it to the final game. After the meeting, the programming department implemented the agreed upon metrics over the rest of the day, so everything was done in a very short time frame. Guidelines for designing metrics were created based on literature prior to the metrics design meeting and used to make sure each metric created was as useful as it could be.

4.1 Metrics and Game Design

Telemetry is the practice of gathering data, and the data collected are game metrics (Drachen, 2012). An example of a game metric could be how many times a user opens the options menu during a play session. To gather this metric, some things need to be defined first. A game session is the time between starting a game and playing until end conditions are met (Mäyrä, 2008). For a sandbox game this definition is problematic, as there are no clear end conditions. For a sandbox game, looking at a play session, the time when the game is actually played (Björk and Holopainen, 2003) provides smaller amounts of data, because the game sessions are very long.

The example of the options menu and its opening is a very simple metric to document. It would just mean that the user has pressed the button that opened the menu. The actual metric that this telemetry would provide is a numeric value of the times the user has pressed the button to open the options menu. “Game metrics are, in essence, interpretable measures of something”, Drachen (2013) defines. One metric can be compared to other metrics, which creates new metrics. Drachen (2013) uses the example on comparing the number of times the player has fired a weapon in a game to the number of times they have hit a target.

While the project discussed in this paper is about iterating on old ideas rather than creating new ones, it did have features which need proof to be considered working as intended. Collecting game metrics was planned to be used in the beta test phase in order to collect data on whether or not the main features of the game were used by the players as intended. Also information about if certain optional features of the game are used at all by players is interesting to help find out if optional features are interesting to players at all. Confirming design choices was the main goal for the team. For the publisher, the main goal was most likely to gain information of what kind of additional features would appeal to the players. This information could then be used to guide decisions on what kind of additional content to order from the developers.

Game design is mainly combining old ideas in new ways and iterating on them (Kultima, 2014). Design work relies mostly on the designer’s expertise and knowledge of games and players in general. When working with a tight schedule and a contract that defines the main features of the game, one has to hope the set features are good enough to provide a meaningful experience for the player. It is also important that there is enough room for modifying them to be more meaningful. Occasionally some systems do not work as intended when they are programmed and tested. When starting a project, the designer has to research other games related to the genre, choose what systems would work together and have enough content make a game in the wanted price range. After that, one can only hope the design is accepted by a publisher and can be scheduled to fit the budget. All of the above relies on the skill and experience of the designer, and the rest of the project depends on it. Most early design work is simply guessing things and hoping to get them right enough in a way that can be later fleshed out and modified during the project.

Having a way to test the ideas and choices during the project before launch in other ways than regular quality assurance testing is very valuable. It can help avoid many problems that are expensive and hard to fix later. Quality assurance testing is a great way to find problems with the game early on, but it is expensive and requires constant supervision and providing enough information with each game version so the testing department knows which parts of the game to test and which to avoid due to them being too unfinished to benefit from feedback.

Some game design is based entirely on game metrics. Social games, like *Farmville* (*Zynga*, 2009) or *Mafia Wars* (*Zynga*, 2009), two successful *Facebook* game titles, are designed based on metrics. *Zynga*'s lead game designer, Brian Reynolds (2009) talks of how *Zynga*'s main strategy was to analyse metrics and create game content based on the findings. The main point of metrics analysis was to find out when players were invested in the game and spent money on it. For free-to-play games, metrics are very important due to smooth learning curves, and because customer commitment is the only way to make a profit.

The way social games use metrics as design tools is mainly in introducing new features via A/B testing. This means that when a new feature is designed or a new game released, two different versions (version A and version B, which give the practice its name) are given to separate player groups (Cheshire, 2012). The groups' reactions after playing are measured via telemetry and the metrics are compared in order to find out which version keeps players playing longer and spending more money. In the *Wired* article by Cheshire, *Wooga*'s Lead Game Designer says that "Wooga's users don't just play a game, they design it". *Wooga* has produced many successful social game titles, such as *Magic Land* (2011) and *Happy Hospital* (2010), and was the fifth largest game company on *Facebook* in March 2014, measured by daily active users. This can indicate that their formula for making social games works well. It can be debated if users really design games, or if is it just a case of using their data to create a profitable game system, but all in all user actions have an effect on the final game.

A/B testing is a fairly new practice in the games industry, no mention of it can be found before Wooga. The practice relies on a large user base, good metrics design, talented analysts and the so called "forever beta" phase. Forever beta simply means that the game is being worked on and iterated continuously. A traditional game development process sees the product start out at an alpha stage, where just basic features exist, move on through a beta phase when other features are added and the game is polished and tested, and finishing up with a release candidate. After the release candidate meets the requirements set for it, the game is released. Later, the game can be patched to add content and/or fix problems, and additional content can be added in the form of expansions. In a forever beta game, the phase where fixes are made and things changed based on feedback from testing, developers, analysts or users never ends.

For *Skylines*, metrics are not a big part of the design. The design is not based on metrics, but some room is left to make fixes to the game later if problems are discovered by analysing metrics. In *Skylines*, the hope is that metrics will provide information that allows design decisions to be confirmed. Additional things that could be beneficial to know include popular player strategies and how smooth the learning curve of the game is. For metrics to have been used as a big part of game design, the game would have needed metrics support from the very start and a group of players or testers to generate data.

This is absolutely possible even for single player offline games (Georg Zoeller, 2013) as was shown in the article concerning how *BioWare* used metrics with *Dragon Age: Origins* (Electronic Arts, 2009). Most of the metrics were collected from in-house testing that was done by developers when working on other things. Metrics proved to be a very useful tool in documenting bugs and making sure information about problems is actually saved and handed over to someone who can fix the issues. Traditionally, if a problem is encountered while a developer plays the game still in development, they have to fill in a bug report. Bug reporting can be done with systems especially created for it, but small studios use all kinds of things. *Colossal Order* has previously used post-it notes (write the issue on the note and take it to the table of the person you think is the one who should fix it) and a wiki (write the issue on a wiki page according to the instructions written there).

The advantage of post-it notes is the easy interface and that they can be written while playing without exiting the game. The downsides were that notes can get lost, there is no guidelines to documenting bugs and other issues, so some notes did not have all the information needed to investigate fixes. There is also very limited space on a note, making larger issues need more than one note, which then increases the possibility of something getting lost or the notes getting separated.

The wiki is fast and easy, but sometimes necessary information was not recorded even when instructions were written carefully. Towards the end of a project the number of reported bugs would have become too big to handle, and dividing them to different subpages would have made navigation hard and time consuming. Also generally the more people are in a hurry, the shorter reports they write and the less they follow instructions.

A bug reporting software makes sure all the needed information (version number, the person reporting, the person handling the issue, requested fix deadline etc.) is inserted in the form before it can be sent. The interface is somewhat intimidating, with lots of drop down menus and different fields to fill in. Filling a bug report takes some time and requires being familiar with the software. It can also be frustrating for a developer who is in the middle of something and would just like to make a note of a small issue and does not feel it merits a real bug report.

Telemetry would automatically track all important information and save the developers the trouble of filling out complicated forms. But for a small team (Colossal Order had eight people when *Skylines* was started) it is too much work when compared to the benefits.

4.2 Introduction to City-builder Games

The best known city-builder game is also the first one: *SimCity* (Maxis, 1989) by Will Wright. The main feature that made *SimCity* different from other games was that instead of winning conditions, it had continuous play. The player does not win, but rather plays in a sandbox for as long as they like. Rollings and Adams (2003) place *SimCity* in a category called “construction and management simulation”. The gameplay revolves around managing a city and building services. Residents of the city have free will: if the player treats them well, they are happy, but they can also be dissatisfied and leave the city. While the game cannot be won, it can end in bankruptcy.

SimCity hovers on the edge of being more like a toy than a game, but it has been marketed as a game and is established as one. If you follow Chris Crawford’s definition of a game, presented in *Chris Crawford on Game Design* (2003), *SimCity* would not count as a game. Crawford requires games to have built-in goals, and since *SimCity* lacks these, in his system it is considered a toy. One could argue that *SimCity* does present goals by guiding the player towards building a successful city and by punishing for bad decisions, but this element is very light. Only a few basic elements of the game must be done and everything else is up to the player’s choice. Salen and Zimmerman (2003) have a different take on the boundaries of games. Their statement says:

“A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome.”

The definition is much more loose, and the player-set goals of *SimCity* can fit into it. The player is in conflict with the game world, trying to reach a chosen goal when the game system demands attention to other things. The question of city builders as non-games comes up occasionally, but they are an established genre and no consumer would feel it weird to have a city-builder game on the shelf of a game shop.

SimCity created this game genre and continued its success with many sequels. At the time of writing, the latest addition to the series is once again named *SimCity* (Electronic Arts, 2013), published in late 2013. The game faced challenges because of its always online design and small map size. Metacritic shows that currently (8.3.2014) *SimCity* has a score of 64 out of 100 based on 75 critic reviews. The user given score is very low, only 2.1 out of 10 with a large majority of the reviews being negative. I would think this negative response from players is due to problems with the always online feature. When the game was launched, the servers could not handle the amount of players, leaving many eager customers disappointed. Even now, months after the launch, people are occasionally unable to access the game due to server problems. This issue is being handled with an upcoming offline mode that allows players to enjoy the game without connecting to the internet.

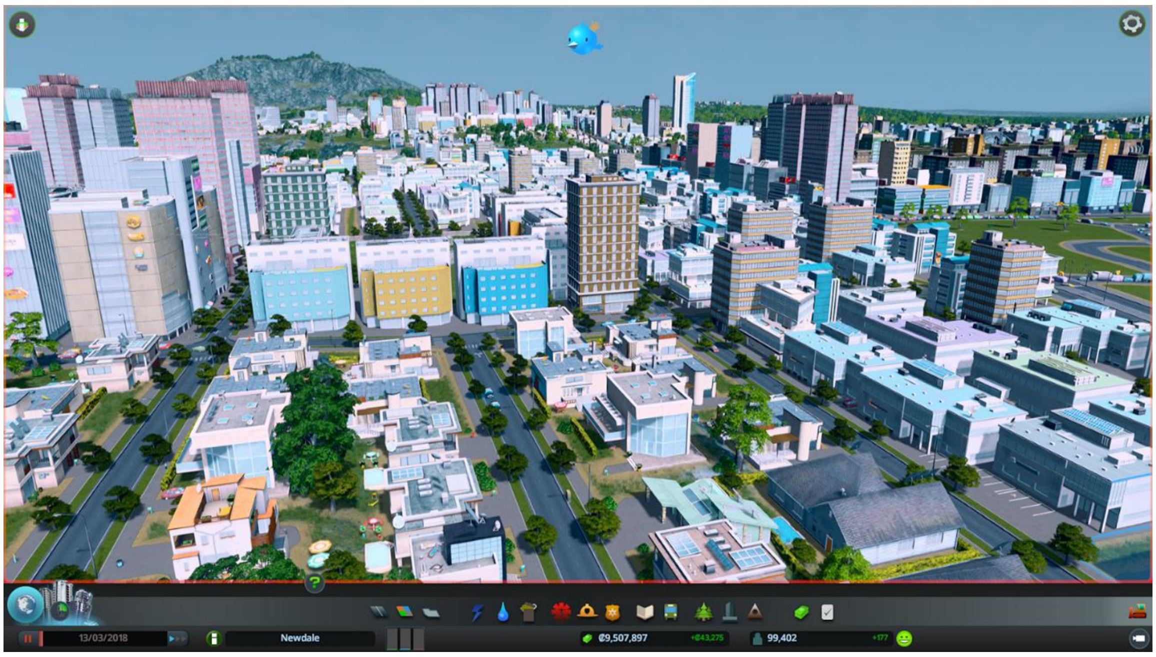
Another well-known city-builder game besides the *SimCity* series is *Caesar* (Sierra Entertainment, 1992). *Caesar* has a slightly different approach to managing the city than *SimCity*. The player is the mayor of a Roman city and needs to provide the citizens with food and goods, and a budget for wartime. Choosing a historical setting and including warfare in the game were the main differences when compared to *SimCity*. The *Caesar* series developed further to even include managing and directing troops, and also got a campaign mode where the player was given goals. When goals were met, the player gained access to a new map (or was taken back to their biggest map where the main city was) with new challenges. *Caesar* spawned a whole series of games called the *City Building Series*, published by *Sierra Entertainment*. They were set in different historical eras, for example ancient Egypt and Greece. While other settings worked fairly similar to the Rome of *Caesar*, they had specialities, like the annual floods of the Nile in the *Pharaoh* (Sierra Entertainments, 1999) or the emphasis on pleasing the Greek gods in *Zeus* (Sierra, 2000).

SimCity and the *City Building Series* have a clear distinction in playing styles. In *SimCity*, the city reflects the player’s decisions, and rather than there being only one way to “play it right”, there are choices for the player to make. The *City Building Series* has a clear right and wrong way to play, and the games in the series have clear winning conditions. While the *City Building Series* games do offer continuous play, the main emphasis is on the campaign that sets distinct goals for the player. Going back to Chris Crawford’s definition of a game, the *City Building Series* is a game, but *SimCity* is not. One could say that *SimCity* is more sandbox-like than the *City Building Series* with the amount of choices it offers for the player to express themselves.

There is also a difference in the economy simulation, so light-heartedly the *City Building Series* could be called communistic. The player owns the whole map and places production facilities to benefit from the natural resources available. They also place homes for workers, controlling how much workforce is available. While it is important to fulfil some needs of the workers, like healthcare, everything aims to increase production and for the player to produce more money. *SimCity* only allows the player to zone land, not actually place homes or production facilities. The main goal is to gain money and build a city, but the player does not own production chains or handle usage of natural resources the same way as in *City Building Series*.

4.3 Cities: Skylines

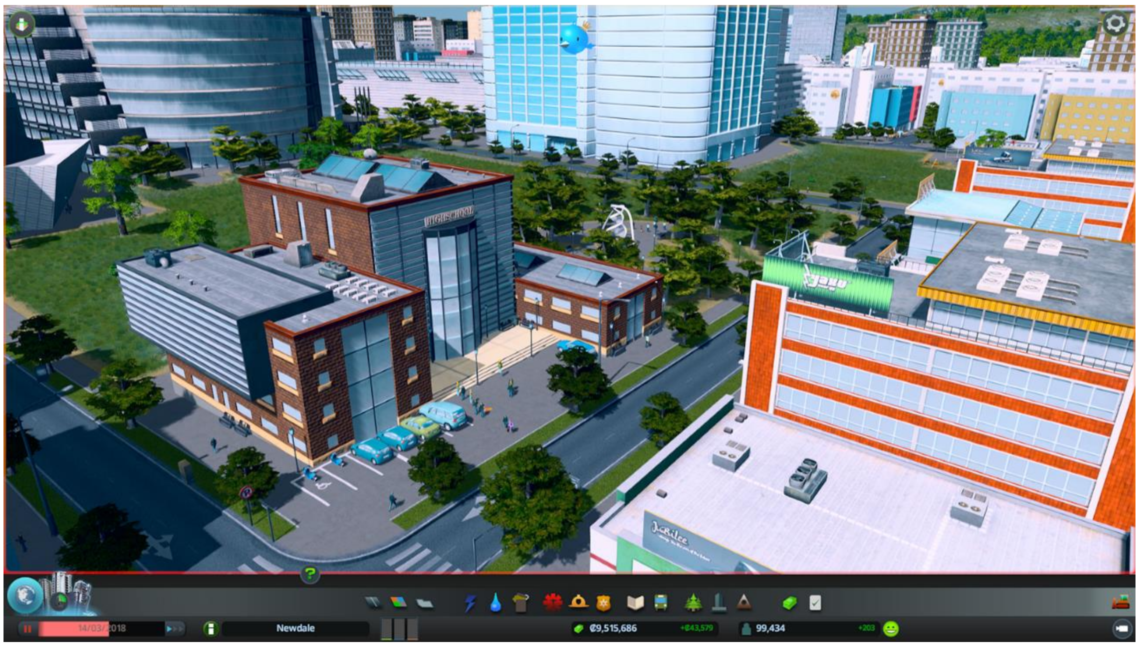
Cities: Skylines is a classic city-builder game. It looks upon the mechanics of *SimCity 4* (Maxis, 2003) for inspiration, but turns towards *SimCity 2013* (Electronic Arts, 2013) for visual presentation and usability. In *Skylines*, the player is in charge of building and managing a city. The citizens are individual entities that have wants and needs, and different moods cause different needs. The core of the game play is to observe and study the city to find out, via visual clues and/or information overlays and messages, what could use improving, or if there are problems or new opportunities. After observing, the player builds or manages to handle the problem, then again observes if the problem is corrected and what kinds of effects it has on the city. Ideally, addressing one problem always pushes the city forward and creates new demands or unlocks new content. The player tries to keep the city in balance, but the actual playing is fine-tuning the city and reaching goals to unlock more content. The game system always tries to keep the city slightly off-balance to always provide the player with meaningful tasks.



Picture 2. *Cities: Skylines.* User interface.

Skylines is challenging but not overly difficult. The player is helped along if things go wrong, and the city has systems that prevent "death spiral" mechanics. A “death spiral” is basically a situation which the player can not recover from: When things go wrong, the player is punished, and for less skilled players the punishment makes it harder to recover so they fail again and are again punished until the inevitable game over comes along or they quit the game because of frustration. *Skylines* has a lot to unlock so it drip-feeds the player, which makes the game easier to approach and caters to less skilled players.

For experienced players, the game offers depth and possibilities to micro-manage in order to get the city working as efficiently as possible. Less skilled players do not end up with optimized cities, but can easily make a city that does fairly well, and they get to see a glimpse of the more complex systems that they maybe get into as they become better acquainted with the game. *Skylines* has lots of internal systems that aim to keep the city evolving and the needs of the citizens shifting, so even experienced players would always have meaningful tasks to choose from and not end up with a situation where the city is as good as it can be and there are no more choices to make.



Picture 3. *Cities: Skylines*. A high school city service building.

The player can build city services, such as power plants to provide electricity, or hospitals to take care of sick citizens and improve the city-wide health. They can zone areas where citizens then build residential houses, companies build industrial plants to produce goods and commercial businesses build shops to sell goods to citizens. All of these zones are linked: residential areas need industry to provide jobs for the inhabitants, and commercial areas to have shopping possibilities. Industrial areas need residential areas to provide them with employees and commercial areas to ship their products to. Commercial areas need shoppers from residential areas and goods from the industrial areas.

In addition to building and zoning, the player can manage the city through taxes, policies and budget. Income is provided for the player via taxes, and managing them means finding a balance between citizen happiness and the amount of money needed to provide required services. The typical way this works is that when citizens get more services, they are happier and willing to pay more taxes, thus covering the costs of the services. Adjusting the city budget allows the user to set how efficiently services work. If they have only a small amount of money in their disposal and do not want to build anything new, they can up the budget for the needed service to make the existing buildings work a little bit more efficiently, increasing citizen happiness. Policies are city-wide “laws” that are used to create areas that have their own character, like a residential area that only accepts senior citizens to live in it.

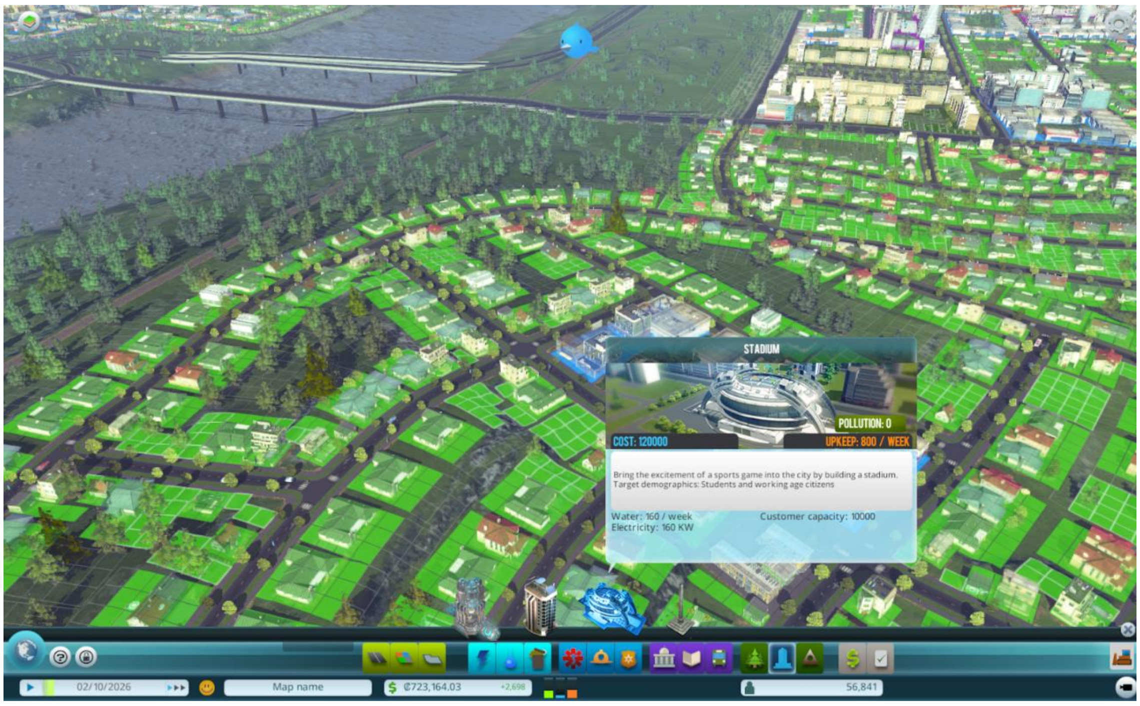
All in all *Skylines* is an old-school city-builder with a polished, easy-to-use user interface (UI) and features that help less experienced players to enjoy the genre but still keep the more experienced players entertained.

All features are not equal when choosing metrics to track. Some features of the game are tried and true in previous games, for example building city services and zoning have been a part of the *SimCity* series for years. We can assume many players are familiar with zoning and placing city service buildings and that these features are generally understandable and fun, and do not need to be investigated further with game analytics. Other features are less common, like the integrated tutorial, so metrics can be very useful in finding out how players react to it and how it is used. This chapter presents the main features that can benefit from looking into how players use them by using game analytics.

Drip-feeding

In order to gradually ease the player into the game, different tools and features are opened slowly to the player. Instead of offering all tools in the beginning, *Skylines* unlocks tools and features as the player grows the city population and thus progresses in the game. During the development, we first tried placing population milestones very far from each other so the unlocking would continue through the game. Testing reported this did not feel fun because services unlocking for a very large city required the player to place many of the same type of service building to cover the whole city. Doing the same placement action over and over reduced it to a mechanic function and according to testing stripped it of the fun. When a new service unlocks, the need for it unlocks at the same time. For example before the university unlocks, workplaces cannot need university educated workers, or before the graveyard unlocks, there can be no dead citizens in the city. These needs are easier to fill and less likely to cause catastrophes when the city is small. Introducing new needs to large cities had the problem that the new need could lead to partial abandonment of the city due to too little services, if the player was not quick to act or lacked money.

Due to testing feedback, we changed the population milestones to be much smaller and thus shifted the feature unlocking to happen at the beginning of the game. According to testing this made unlocking feel more like a tutorial for less experienced players and rewarding to more experienced ones. See table 2 for final unlocking system.



Bring the excitement of a sports game into the city by building a stadium.
Target demographics: Students and working age citizens

Water: 160 / week
Electricity: 160 KW
Customer capacity: 10000

Picture 4. *Cities: Skylines*. User holds mouse cursor over unlocked "Stadium" building in the build menu for Unique Buildings.

Drip feeding is common in games, as it is easily linked to realism and thus easy to explain to players and players take well to it. For example role-playing games such as *Dragon Age* –series (Bioware, 2009-2014) and *Diablo 3* (Blizzard Entertainment, 2012), use character skills that unlock as the player progresses. These tap into the idea that games are "learning machines" (Gee, 2004) that arrange tasks so that the player starts with small, easy tasks and progresses slowly to larger and harder tasks. Also the complexity of the game slowly grows to match and grow player skill. There are games that do not use this method, for example the dynasty simulator *Crusader Kings 2* (Paradox Development Studio, 2012) that starts with the player choosing freely what country to play as on a map of medieval Europe. All features are available to the player from the start, meaning that a huge amount of information needs to be absorbed to be able to confidently play the game. *Crusader Kings 2* has done well in receiving critical acclaim, showing a Metacritic score of 82 on 26.4.2015. Games that do not guide learning are enjoyable to some players. Comparing sales numbers, *Crusader Kings 2* reached 1.1 million sold copies on February 2015, three years after release (Paradox Interactive infographic, 2015). *Skylines*, that uses the drip-feeding method to support learning, reached million sold copies in five weeks. This is not the only difference the games have, as one is a dynasty simulation developed by *Paradox Development Studio* on their own Clausewitz engine, and the other is a city-builder made by an independent developer on Unity engine. Both are published by Paradox Interactive and promoted to "Paradox players", people who enjoy the type of games Paradox Interactive usually publishes. This difference in sales could still mean that it is highly beneficial to use systems that support learning and offer guidance, especially if aiming for large audiences. No tutorial at all or very light support rely on the player having motivation to learn, and in can be enough. For players who are not at first motivated, tutoring and supporting may be the key to getting them invested into the game.

Table 2. Milestone requirements and unlocked content in *Cities: Skylines*

Milestone	**Required Population**	**Areas**	**Features**	**Services**	**Zoning**	**Policies**	**Roads**	**Buildings**	**Money Bonus**
Little Hamlet 1	500	1	Taxes Loans	Education Garbage	—	—	—	Elementary School	20,000
Worthy Village 2	1000	2	Districts Second	Fire Department	Agriculture Forestry	Power Usage	—	Fire House Police	20,000
Tiny Town 3	1300	—	Decoration	Level 2 unique buildings	—	Parks and Recreation	—	High School	20,000
Boom Town 4	2400	3	—	Bus Level 3	Ore	Recreationa l Use	Cloverleaf Intersection	Advanced Wind Turbine	35,000
Busy Town 5	4600	—	City planning policies	Level 4 unique buildings	Oil	Free Public Transport	Decorative roads	Fire Station Hospital	—
Big Town 6	7000	4	Taxation policies	Level 5 unique buildings	High-Density Commercial	Education Boost	—	Incineration Plant	45,000

—	—	—	—	—	—	—
Cargo Train Terminal	Crematoriu m	Solar Power Plant	Cargo Harbour	Nuclear Power Plant	Airport	—
—	—	—	—	—	—	—
Big Business Benefactor	—	—	—	—	—	—
—	—	—	—	—	—	—
Level 6 unique buildings	—	—	Ship	—	Plane	—
—	—	Third Loan	—	—	—	Monuments
—	5	—	6	7	8	9
10000	16000	19000	30000	40000	65000	80000
Small City 7	**Big City 8**	**Grand City 9**	**Capital City 10**	**Colossal City 11**	**Metropolis 12**	**Megalopoli s 13**

5 DESIGN WORK AND ANALYZING DATA

This chapter will concentrate on what was done when designing metrics for *Skylines* and how the data gained through telemetry was analysed.

5.1 Testing and Analysing Testing Results

During the development of *Skylines,* playtests were performed to give information on bugs and what sort of experience it is to start the game. The main reason for focusing on the beginning is that it is the time when users are supposed to learn how the system works and the game systems should help to bring users roughly on the same skill level, so that all players have the necessary skills to play the game. These tests have a script with tasks for the player to complete and playing is documented on a video that shows the face of the tester, the keyboard and mouse, and the computer screen. The playtests were very useful in finding out what sort of problems new players experienced. They also helped a lot in finding out what some more obscure bugs reported by the testing were actually about. Videos showing player reactions and actual use of keyboard and mouse were a great way to pass on information about how the game plays and what problems there are. Sometimes, written bug reports are not very informative and require the developers to test to reproduce the problem, which takes time. Looking at a video of a bug makes it fast and easy to see what the issue is all about, especially when the videos are filmed so that you can see the face of the person testing the game and their keyboard and mouse. Seeing more than just the game screen helps a lot in finding out what the player is feeling; if they are confident or feel insecure.

Five testing sessions were done during the testing. The first test was done when the game was in a very early stage and had only a few main features implemented. The tests were done during development and consisted of 3-5 different players playing for 30-45 minutes, with their complete play sessions recorded. The players had a person sitting in the same room, giving them tasks and helping them out if they got stuck. Participants then filled a survey measuring their feelings of the fun factor of different parts of the game. Based on these tests and separate quality assurance tests, bugs were reported and the fun factor of the game was evaluated. The tests done with video capturing were usability tests with different types of players, all the way from people who had never played city-builders to very experienced players. The purpose of these tests was to observe how different player types played the game and to see how well the unlocking mechanic and guide messages helped players in learning the basics of the game. Later quality assurance testing was done by professionals and meant to weed out bugs.

On most testing rounds, the quality assurance run by the publisher was responsible for creating tasks for the players to see how well the players grasped the mechanics. The tasks where along the lines of “create a city”, “tell of how well your city is covered by police services” or “make a building level up”. Some were very broad, high level tasks to get the player started, and others focused on a specific mechanic to see if the player understood how it worked.

Only during one testing round was the developer asked to contribute to creating the tasks for the players, at all other times this was done by people on the publisher’s side. These tests were very useful; early testing is the key to catching design problems early and trying out different approaches. These testing rounds were not very well documented, as they were more of the type used just for one particular development phase and then discarded when a new test was done. Usually, a list was made of the issues found during the testing round, a meeting was held to determine how to fix the issues and who could do it. Once the issues were fixed (or on some occasions forgotten after other issues took the spotlight) they were marked as fixed on the list and the list was never visited again.

Some problems arose due to a few players being severely unmotivated while playing. This is only natural when tests are done, as not everyone likes all games and some people just do not perform well in test environments. The low number of testers meant that if one or two players out of three or four were unmotivated, there was very little useful information left and their score affected the fun ratings greatly. One of the reasons this became a problem was the fact that the developer had asked for the publisher to use people who like city-builder games. However the developer had not specified what kind of city-builder games. There are two main types, which differ greatly (see chapter 4.2. Introduction to City-Builders for a detailed breakdown), and a player who is familiar with one type could still be very lost with the other one, creating a situation where it looks like an experienced city-builder player cannot grasp *Skylines* at all and this caused great concern. When the problem was noticed, the developer provided information on what specific city-builder games the players should be acquainted with and tests went much better afterwards.

5.2 Why Use Game Analytics for This Project?

The main reason for including metrics in to the game was because of publisher requirements. The publisher provided a list of a few key metrics they wished to document and offered their own servers for use in handling the information. For this project, telemetry was not planned before the publisher's request, mainly due to the small team size. With a limited number of people available, implementing telemetry would take up too much resources and analysing the data would have consumed even more resources. Planning for the analysing phase would also have been difficult, because there would only have been time for it if resources were free after the project, and that would have meant a non-ideal situation. Ideally, all resources from this project would be used for the next project.

For an independent game studio, making changes to the game based on telemetry information is only an option if the publishing contract has been written with that in mind. If the final product is the basis for payment, further changes cost extra and have to be negotiated separately. Using a large amount of resources during production for something that might provide extra deals after the project has been finished is often not something the schedule allows.

After the publisher initially provided the list of metrics they wished to observe, the developer team wanted to look into telemetry further, in order to see if it could be used to confirm design choices and provide information for future projects. This would only be done if it could be implemented with little effort in addition to the publisher requested metrics. The publisher's request meant that the telemetry system needed to be programmed anyway, so putting some extra effort into gaining valuable information seemed like a good choice, because the programming that was the bulk of the work would be done regardless.

The main findings the developers expected were confirming if design choices were right, if the users play the game as designed or if there is possibly some emergent play. With a sandbox game, it is very valuable to know what most players enjoy doing in the game, so additional content can be targeted to the largest possible audience. This was also the intent of the publisher in addition to gaining some general information on their user base. General information on users is valuable to the publisher because the publisher has a very unique fan base and a catalogue of games targeted to their fans, mainly meaning medium to heavy simulation game fans. If a person enjoys one Paradox published game, they most likely enjoy other ones too, because even with different genres, the amount of detail and relation to history are a continuing theme. For a game studio, the information on player behaviour in the game is interesting because correct analysis of the data would allow the studio to offer additional content packs with data showing they really work for the target group.

After the initial meeting on publisher required metrics, it became known that while the publisher did have servers ready to accept data, the system for data retrieval was not yet ready and had no estimate on when it will be ready. This made the team reconsider their choice to collect more than the needed telemetry, because it was not clear if the information collected would be available in a readable form at a time when it would be most useful. Additional telemetry was put on hold until a very late stage when the server system was finished and a promise could be made that extracting data was possible during the late stages of the project.

The original plan had been to have telemetry programmed when a closed beta test would be run before the launch of the game. Information collected during the beta test would be used to confirm design choices and help weed out mislabelled bugs that always are reported during a beta. During the beta phase there would still be time to adjust the game if the telemetry data would show some aspects of the design were not used by players as intended.

5.3 Chosen Metrics

Most metrics were chosen based on publisher wishes. These include the most common metrics, like daily unique users and users that have not started the game in 30+ days. While these metrics were not explained to the developer further, it is clear they are used to find out player numbers and fluctuations in player population. Previously there have only been sales numbers, which tell of how many units of the game are sold, but with no metrics collection it has not been possible to tell how many people actually play the game and in what kind of patterns.

5.3.1 List of Publisher Requested Telemetrics

The following indented parts are quotes from Colossal Order's closed wiki used to store game information.

> Telemetry requested by the publisher
>
> - DAU: Daily unique online users starting a game.
> - MAU: Monthly unique online users starting a game. Tracked daily and includes the last 30 days from that date.
> - (Stickyness = DAU vs MAU ratio: The daily "stickyness" of the game. Overall engagement of the userbase. Higher DAU/MAU ratio means more likeliness of users spending on DLC or microtransactions.)
> - New Users: New users starting their first game that day.
> - Reactivated Users: Users starting a game that day that have player at least once 30+ days ago.
> - Active Users: DAU-(Reactivated+New Users)
> - Churned Users: Users that day that haven't started a game in 30+ days.
> - Playerbase Growth: Users in - Users out daily. Out are "Churned Users". In are "Active + Reactivated Users". For growth this should to be in the green as much as possible.
> - first_launch - First time the game is started
> - start_game - Game is started
> - exit_game - Game is terminated
> - meta - unique comp/device identifier, comp description, graphics card model, renderer/driver, video memory, supported fillrate
>   - meta - highest supported shading model, installed operating system, operating system language, CPU specs, amount of cores, system memory
> - login - User logs into Paradox account
> - create_account - User creates a Paradox account
> - start_session - Play session loaded
> - end_session - Play session ended

- custom_content - On custom content loaded
- error - On intercepted error
- unlocking - Milestone reached

5.3.2 Developer Chosen Metrics

The developer had never worked with telemetrics before. Since several members of the team found them intriguing and wished to gain additional information by collecting more than just the publisher requested data, some room was made in the schedule to allow for a day of designing and implementing extra metrics collection. With previous projects, we had made patches to the game to fix some problems and the same was expected to happen with this one, so any extra information on usability problems, their true nature and possible fixes would be useful. If there had been no possible use for the extra metrics, no time would have been spent on them, but in this case the team agreed that they could prove useful and there were people interested in developing them.

Our way of using this day was to first have each interested person write notes about what they thought would be valuable and what sort of limitations were involved. We then sat down to discuss, with the intention of coming up with clearly defined metrics that are within the scope of the time given, so they can be implemented. What this meant was that as many of the metrics collected as possible should be things the game system already collected. The game already had built and planned systems that collected data, for example information overlays for the player to use that list things about the game (traffic information overlay shows how many vehicles are currently on the city streets and what are their types), a general city happiness icon (a smiley face that is derived from the average happiness value of all citizens in the city), or a statistics screen shown as a reward when the player has unlocked everything (graphs describing city development).

For the meeting, guidelines were formulated to guide metrics design based on how Drachen et al. (2013) describe starting out with metrics collection. The guidelines were descriptions of what information was hoped to be gained with the extra metrics. These were the guidelines brought to the meeting:

1. How to find out player strategy

2. How to find out if a specific feature works
3. Confirming design choices
4. Helpfulness of the ingame guide (does the player use it?)

Guideline one, “How to find out player strategy”, is trying to focus the metrics so that by analysing them it would be in some way possible to find out what sort of playing strategies players use. These would include, for example, leaving the game to run overnight, playing intentionally slow to gain lots of money, rushing through the unlocking phase or just being plain lost. If we could identify player strategies and see which ones are used most, it would be easier to balance the game to work for those strategies, or if some strategy seems too easy it could be made harder.

The second, “How to find out if a specific feature works”, guideline is about isolating features and trying to find metrics that would tell about specific features and how they are played with. This guideline was mainly written with the system of Unique Buildings leading to Monuments in mind, but can work for other features too. Unique Buildings are sort of in-game achievements that the user can choose to strive towards. A requirement for gaining a Unique Building could be to have 500 000 citizens or to have average ground pollution in the city on a certain level. They are goals offered to the player to support sandbox play, there is no need to gain any Unique Buildings, but they are things the player can choose to go for if they feel the need for a goal. Building a set of Unique Buildings allows the user to build a Monument, which is a great, visually stunning structure that fills one of the needs the city has. For example the Fusion Power Plant produces so much electricity that it basically removes the need to produce electricity for the city in other ways. This is meant to be a point where the rules for the game change to encourage the user to keep playing with the Monument in the city.

“Confirming design choices” is the third guideline and focuses on metrics design that would help confirm design solutions for things that are less tried and tested in the game. These would include the Unique Buildings and Monuments mechanic, drip-feeding by unlocking and the integrated tutorial.

Guideline four, “Helpfulness of the ingame guide (does the player use it?)”, is focusing on the integrated tutorial. Usually, city-builder games have a separate tutorial, but since we decided to work on an integrated one instead, seeing if it works as intended would be very interesting. Most game tutorials feel quite separated from the actual game, and users dislike them (Glajch et al., 2006) partially due to feeling they need to wait before getting to the actual game. Tutorials can lower the enjoyment players get from playing the game even when the skills of the player and efficiency of playing the game does rise. Our version is a collection of guide messages that only pop up if the game system notices the user does not seem to grasp a part of the game. This is not something you see often, and the reason could be that integrated tutorials simply do not work. We did some research and wanted to try anyway, because nothing seemed to imply integrated tutorials were automatically bad or did not work.

After discussing the guidelines, the meeting proceeded to presenting notes and deciding what metrics could be collected without too much extra work and would provide information fitting the guidelines. We decided to write clearly on each metric the information we hoped to gain. This was according to Drachen et al. (2013, 31-35), hoping to avoid redundant metrics and metrics that would just collect clutter. We wanted to keep the metric number fairly low so analysing metrics later would not be too big of a task. At this point of the project, it was impossible to know when any actual metrics data would be available and in what phase the project – or a future project – would be at that point, so expecting there to only be little time for analysis was playing it safe.

5.3.3 List of Developer Chosen Metrics with Descriptions

The following indented parts are quotes from Colossal Order's closed wiki used to store game information. These metrics were designed by Colossal Order, the developer, not requested by the publisher. The lists are the actual metrics designed for the game, written over the course of one long meeting.

Building

Building an item

1. Information wanted: Seeing if there is a point in the game where building does not happen any more or a point where the user has to place lots of buildings suddenly. New map tiles should show. Also usage of items should tell which ones are the player's favourite ones (for example trees and benches).
2. Stat manager tracking of buildings placed, timestamp and item name, per map played. instead of continuous telemetry use stat manager to track the data on a map session.
3. Also includes the amount of the same items in the city
4. Unique id for each new game started, not additive but tied to new game IDs. going to main menu ends a session.
5. Timestamps can be stored for each placing time but it means lots of data traffic. If stat manager sends all data at the end of the session, the timestamps need to be created. Ideal send per 2 minutes or so.
6. If a building is deleted it has the info on how long it has been on the map
7. Game session individual ID
8. Roads send amount of segments built and when deleting remove the amount of segments to keep the count matching
9. Buildings abandoned
10. Buildings burned down

The Building an item metric tries to find out if there are building related activities for the player to do throughout the whole game. We are expecting to see the event of a new map tile unlocking to produce a spike of building. Listing the number of each building placed will tell if some are used more often than others, for example if a police station would be underpowered, the player would need more of them compared to other buildings. This would then show in the data. For small items, like trees and benches, we could plan extra content based on what types are most popular and making more of those to offer variation.

Because this was the first metric written down, it also has notes on how the telemetry system will work technically. The game already creates a file that is used to update data in game systems. Some of the data on that file is also saved to another file, which will be sent to the telemetry server at set intervals. Creating accurate timestamps would have created lots of traffic that would not have accomplished anything. Members of the programming department came to the conclusion that saving data related to player actions every two minutes would be enough to create usable timestamps and would not place too much stress on the system.

There was also the issue of identifying separate maps and game sessions. The game already had a system to track things related to each individual map session, so progress towards Unique Buildings could be tracked correctly. We chose to attach the new game identification that marks each started game as its own to the building metrics, so we could see data for whole sessions rather than having the noise of, for example, a player starting ten new games in one hour and placing a water pump building in each.

Road sections got just a light version of the building metrics, since there are so many roads in any given game session. By looking at the number of segments and their increase or decrease we hoped to see how the players expand their cities, for example if the roads and building increase almost at the same time, meaning players build roads and quickly add services, or if they first build road network and then gradually add buildings.

Demolishing

Demolishing - when player demolishes something, must separate zoned buildings and player placed buildings

1. Information wanted: same as in above (Seeing if there is a point in the game where building does not happen any more or a point where the user has to place lots of buildings suddenly. New map tiles should show.)
2. Use the same mechanic as building metric to track data
3. Both can be used for any items in the game

To gain information about demolishing, we decided to use the same system as for building. Demolishing would tell us if players upgrade buildings by demolishing old ones, or if they move parts of the city at certain points of the game, which might mean they need more tutoring in the beginning.

Milestones

Playtime to complete population milestone

1. Information wanted: Are the milestones paced well, is there a place where users drop the game.
2. Real-time can be done with already implemented stuff: start new game, timestamp, for return to main menu timestamp, unlock happens, time before unlock event with the same GUID is the time taken to reach milestone. Can match more than one of the same unlock event because the GUID names them.

Playtime should tell us if milestones are paced well. The expectation is that the time to reach the next milestone grows constantly. Tracking unlocking can already be done with the Game Unique Identification (GUID) tags combined with tracking unlock events.

Speed of the game

How much time the player spends in pause and different speeds

1. Information wanted: Is the pacing good.
2. Finding out if the pacing is good: If there's a lot of time spent on pause, there's pressure, if a lot of time in fast forward, the game is too easy. Is the simulation speed as designed. Is performance good.
3. Real-time is more important because the simulation speed can vary with different hardware and map size
4. Buffered and sent a few minutes or so, like the building information
5. Requested speed and real speed can tell if the speeds work as the player wants. is normal speed too slow? is medium speed usable?
6. For game pacing purposes, use only data from setups that run on designed speed, disregard sped up or slowed down setups

The game has different speeds for the user to choose from, because sometimes it takes time to accumulate money or citizens. There are many things that game speed can reveal. We already knew that the simulation speed slows down if the machine playing the game is running out of memory. When the metrics collection was planned, the game did not have low definition models for any 3D objects, which severely affected game speed, but this was expected to improve greatly when the low definition models were added.

The expected information based on game speed would be that if users spend much time on pause, they might feel the difficulty level is too high. If many users play the game on fast forward all the time, the game is possibly too easy. There could be something wrong with how long it takes to perform and action on a certain speed, for example normal speed could be too slow for any enjoyable playing, which would force players to use one of the higher speeds. These are things that could be patched to work better later on. They do not require much work to be fixed, but a huge amount of testing would be needed to decide if a speed is nice to use.

For analysing if pacing works as intended, a system was created that detects if the simulation speed is as designed. If not, it will tag the information so it can be excluded from analysis if needed.

Speed of the simulation

Speed of the simulation reveals performance issues

1. Information wanted: Performance issues affecting simulation speed, the severity of these issues.
2. The speed of simulation can be tracked and can reveal performance issues. Needs perhaps too much work: Median, average, min and max. is reliant on other processes running on the computer.
3. Steam stats tell there's a lot of two core computers. Does this match actual users? Game already checks the computer stats on startup so even changed setups can be tracked to show performance.

For possible performance issues we wanted to give some additional emphasis to the simulation speed metric. In the past, compatibility testing arranged by the publisher has been very light, resulting in problems when some computer setups do not run the game as they should. By tracking the median, average, min and max speeds at which the game runs, we could get a picture of how well the system requirements work. We can also match player setups and their performance, so we could be able to pick out problem setups to request specific testing on, or even come up with a fix that helps the specific setup work better.

Registering the game

When is game registered - time played before registering happens

1. Information wanted: Comparing this data to other data can tell of possible problems. Many times people register when they encounter a problem and need tech support. If

people register when parks become more used in the game, the reward for registering needs to be something used earlier in the game.

2. We track steam IDs and can find out how long the user has owned the game before registering.
3. Not absolutely valid, because people can register from outside the game. Registered user can keep from never logging in from the game.
4. Users are given a reward to encourage them to register.

The publisher is very interested in getting players to register with their own user account system. This helps to collect more specific information about users and possibly to market other titles to them. The publisher's account system is tied to their discussion forums, which also have the technical support section, so having an account is actually quite beneficial for the player. They can get support fast and easy with an account if they run into any problems. Also having information about bugs and problems on the technical support forums is a way for us as developers to learn of user experiences and what sort of difficulties they are having. Encouraging the users to have accounts in the publisher's system helps everyone. This is why we wanted to gain more information on whether or not our users register and at what point of the game it happens.

If the in-game reward (users who log in to their account from inside the game get an extra building that just offers more variation to the game) is the main thing that gets people to register, there should be a spike when reaching the milestone where the Parks & Plazas unlock. Before that point the user cannot do anything with the reward.

If running into a problem gets the user to register, there should be no clear spikes, unless there is a bug in a particular asset or some point of the game is not working correctly.

We also noted that this metric will never be absolutely reliable. People with accounts can choose not to log in from within the game and users can register outside the game too. This needs to be remembered when analysing the data, so it was specifically marked as such.

Unique buildings

When and how many monuments are placed compared to how many are unlocked

1. Information wanted: If monuments are interesting to the player.

2. We can see what has been unlocked in monuments too
3. Can be cross referenced to buildings placed
4. Timestamp tells when these happened
5. Covered already

Monuments have been later renamed to Unique Buildings. The section above talks of Unique buildings, but using the old name because it was written prior to the name change. We noticed that our previous metrics covered this metric already, thus it was marked as “Covered already” on the notes. Looking at unlocking events tells us what unlocked and when, and this be compared to placed buildings, to see if the user places the unlocked buildings as soon as they unlock. These two metrics together should also tell if there are Unique Buildings unlocked and if they are placed on the map or not.

Frames per second

Frames per second

1. Information wanted: The general performance of target group's computers.
2. Needs to show average, peaks and lows. Needs a lot of work to provide useful information.
3. Average can show if users have slow computers, can help with the next project to give info about user base.
4. Only stored in the game, not menus.
5. Only average to lessen workload.
6. Counting done after loading.
7. Time per frame, because can be used for measuring time internally.

Frames per second was chosen to be tracked mainly to learn more about our user base. There was previously no information available on what kind of computer setups our target audience had, so designing system requirements has been done based on general knowledge and experience. This has worked well enough, but could be improved with actual data instead of guesses.

Load time

Load time

1. Tells of how long loading times are, which tells of performance.

Load time is a simple way to look at the performance a user’s setup has.

Population

Population NEW

1. Can be used for marketing (“20 billion citizens moved in to Cities: Skylines!”)
2. Can be used to check if game is working as designed, a steep dip means there's a problem.
3. Can sum sessions with the GUID or use the newest ID
4. To get a sum, choose GUID totals of a given time.
5. Difference between last time collected and current time stored.
6. Store last population and new population, these reveal the change.
7. Current amount of population, delta since last time, actual delta shown to user all stored.

The population metric was added later to the list, thus it is tagged “NEW” and is lacking a description of what information should be gained with this metric. Population metric is yet another metric that should offer information about whether the learning curve is smooth or not, if city services work nicely and if unlocking them creates bumps that might mean the player is not enjoying the game as much as they could. By looking at population, we could see what the usual sizes of user built cities were and how long it took to achieve such population numbers.

Consistent dips in population could mean something is wrong with a city service, which is possible because the need for a service unlocks at the same time as the service does. The population tries to stay fairly stable if nothing is hindering it, but if a service fails, it creates abandonment and thus lowers population. If, for example, the garbage service stops working because landfills are full and the player does not react, it will cause garbage to pile up, which in turn will make citizens eventually abandon their homes. Abandoned houses can be re-built if there is demand for the zone type they are on, so the population would recover, but then the abandonment would reoccur if the original problem were to persist.

During development, service needs appearing when a service is unlocked had caused problems. For example, if the player had proceeded very slowly, the unlocking of a way to take care of the dead caused a death wave in the city, because a sizeable number of citizens had grown to be seniors but had not yet died simply because dying was not possible before the way to handle it became available. These kind of problems are most likely caught during testing, but if something should slip through, a population metric would reveal it easily. This kind of a problem would show fast on the forums, but with actual metrics data we could get more information on what is actually happening instead of relying on reports from individual players.

Money

Money should be tracked just like population, but with no delta (no relation between new and last)

1. Highs and lows are the most important when looking at data
2. Current amount of money tracked and compared to last amount saved
3. Designing maximum six metrics per request helps with server queries when trying to analyse data. Up to six can be stored in one message to server, which is a limitation from Paradox servers.

Money was another thing we thought could help identify places where the game works or does not work like it should. The database used for storing the information has a limit of only having up to six different values in one query. Having three different values regarding money (current number, compared to last time saved, what is actually shown to the user) works with this limitation. The number of values came up when designing this metric and was written down when the subject came up in the discussion.

Integrated tutorial

Ingame guide

1. How long is the message visible before the action required is performed?
2. Placing a building spawns a reminder pop up message, if the message is ignored, we can try to find out why.
3. We can compare with money to see if user can afford it.

4. How many times a message is shown? Guide now counts how many times a message pops-up, so that can be tracked. Tracks per saved game using the same ID. (Wishlisted, needs too much work and is separate from the others)

The integrated tutorial – here called the in-game guide, because we tried to avoid calling it a tutorial officially so as to not scare off people hating tutorials, and because it also actually works more like a guide than a typical tutorial – is mentioned separately here, because it is a feature that we constantly worried about. The guide is quite a complicated system, so in the end we did not come up with any decent metrics to track if it works like it should. The time a message is visible can be tracked but does not give much information. A message can be left open because it is ignored or because the user is trying to perform the action they think will resolve the problem presented in the message. The second and third bullets ended up being redundant, because we later on decided to have the reminders not show if there is not enough money for building. The reminder is a message that pops up when the user has unlocked a building but has not placed it for some time. It was very useful for making sure users notice new services becoming available, but was annoying if it popped up when the user had no money to fix the problem.

The fourth parameter would have been useful in finding out the types of messages that were most common. This, however, was so separate from all the other metrics, and the system had no built-in way of tracking the metric that it would have been too much work and could not be fitted into the one day schedule. It was left on a wish list so it could be done later if there was still time left. Most often things on the wishlist never get done, so basically we just had to accept the fact that we could not come up with a way to get usable metrics regarding the guide messages.

5.4 Expected Metrics Data

Based on experience and examples from other city-builder games and games in general, there are expectations of how data should look when things go right. Some data should show clearly if there is something wrong with *Skylines*. This chapter lists some of the expected patterns in the data if the game is not working as it should.

Building an item

If building an item works like it should, it should show that players' cities have varied service buildings and they are built in bursts. Most likely in the beginning of the game the user builds all new available buildings when they reach a new milestone. These spikes become less apparent if money is scarce, as the user will have to wait for money to accumulate before building. When all buildings have unlocked, new map tiles and building new neighbourhoods on the existing tiles should show as spikes of building action.

A very low amount of a particular service type is a sign of a problem. It would indicate that said service is overly effective, and that individual service buildings should be made less powerful or, inversely, that all other services need to be boosted (their effectiveness increased) to match the more powerful ones.

Demolishing

Spikes in demolishing could mean that some service buildings that unlock later in the game encourage the player to demolish old, small service buildings and replace them. A guideline in the design of *Skylines* is that the player should not have to demolish work they have done, so seeing spikes means the smaller service buildings might need to be made more efficient or visually interesting to make players want to keep them.

Playtime to complete population milestone

This should be a rising curve. The first milestones are very fast to meet, then they should continuously take longer. Spikes or dips mean that the requirements for population milestones need to be modified one way or the other. Finding the right numbers for this feature had been challenging, so the curve would help find the last spots that need tweaking.

How much time the player spends in pause and different speeds

Time spent on pause, based on my experience and expectations of play styles, should be something around 15-20% of the whole session. If the metrics would not support this, the first thing to check would be if my guess was wrong instead of immediately trying to fix the game. Asking about play styles on the game's forum might be a good way to check if the expectation was wrong, but it should be kept in mind that the active forum users might also be the more enthusiastic players, who spend more time on pause, optimizing their performance. Despite my efforts towards finding some, no literature on the division of playing paused versus non-paused seems to exist. In many games, pause is also a state where the player can do nothing. In *Skylines*, pause was designed to actually only be a paused state of the simulation. The player can still place buildings, browse menus and open and close information overlays. It was designed with the intention that, when faced with difficulties, players can pause the game and think about their next move in peace without the threat of everything going downhill if they do not instantly come up with the right course of action.

Speed of the simulation

Simulation speed is used to find out if there are performance issues on certain setups. The aim is to have the speed remain mostly the same throughout the game, but slight slowing down on large cities is acceptable. Big drops can mean there is a memory leak or a 3D model that is missing its low definition version.

When the game is registered

If registering takes very long, there can be a problem in the way it is suggested to the player. Ideally players would register early, within a few days of starting the game.

When and how many unique buildings are placed compared to how many are unlocked

Ideally, all or nearly all of achieved unique buildings should be placed. If there are many buildings that are unlocked but hardly any are placed, there might be a call for them to give the player better benefits so as to be more attractive as a player chosen goal.

Frames per second

If the average fps is over 60, the game is performing as it should.

Load time

Ideally, load times should be at maximum one minute. If there are much longer loading times on some setups, it could be possible to optimize the game to work better on those.

Population

Dips in population would hint that some city service needs appearing can create problems the users are not ready for. The game has a safety feature that if something in the city goes wrong, there is no game over even if all the buildings of the city would be abandoned by citizens, but zoned buildings suffering from the problem get abandoned, then can be re-built, but will get abandoned again if the problems persists. This “safety abandonment loop” communicates to the player that there is a problem, but allows as much time as the player needs to handle the problem, allowing the city to then recover without the need for the player to bulldoze large areas. The service needs creating waves of abandonment was seen a few times during development, so was an important thing to keep an eye out for. Ideally, population should always stay almost the same or rise. The “safety abandonment loop” creates recognizable patterns in the population which help to recognize if the players are having a hard time at a certain phase of the game.

Money

It usually is not good when the player suddenly has immense amounts of money at their disposal. This would show on the money metrics. The game has a system in place that slightly lowers the income based on the amount of money the city has, so that when the user has plenty of money, they will not gain more as fast as they would with a smaller sum saved. This feature had not been tested much, so was possible that something might go wrong with it, or its effect might be too light and not slow the flow of money at all. If everything worked like it should, the average amount of money should grow a little during the game, but there should be steep dips where the user purchases a new map piece or constructs a new neighbourhood.

5.5 Practice

For Colossal Order, when working with previous titles, main sources of feedback have been the publisher’s discussion forums. Each game has a general forum and a dedicated tech support forum that is only unlocked for users who have registered the game via the forum system. The general forum has discussion of all things relating to the game, like playing strategies, exchanging hints & tips on how to play, adverts for player created communities around the game and player made content. The tech support forum is only for reports of problems and there is personnel answering the questions as best they can. Major bugs that have not been caught by testing runs during development have been found on the tech support forum. This has meant the team has had to spend time browsing the forum and waiting for community manager feedback on issues that need attention. Community manager is the person responsible for replying to users on the forums and writing official posts of information the players need to know, for example a post with a list of all new features and fixes a new patch brings to the game. The community manager often also collects the most discussed topics and repeatedly appearing bug reports for the developer and publisher, so the issues can be fixed.

The general discussion forum has been a guide for what sort of additional content the players would like for the game. Threads have been started by the developers to specifically ask users what they would like to see, and the community managers have made sure there is always a thread about suggestions and ideas on the front page. This makes finding information easy, as it is mostly in one single thread of forum posts. When we developers have asked for ideas, the system for analysing the results has been to simply count what has been mentioned and how many times. We have also done polls, which removes the extra effort needed to go through the answers. While we do listen to the users, naturally not all suggestions are possible. So, after coming up with a list of suggested things, they are rated for cost mainly based on workload.

Polls have also been used to involve users in the development, because at least the ones who are active on the forums seem to enjoy that. If it has been decided that an expansion will be, for example, a pack of five new vehicles, the users sometimes got to vote which vehicles would be included. Many people answering the thread mentioned that they had favourite vehicles or wanted a specific one added so they could simulate a certain real-life city better.

There is no way metrics could replace the polls and idea threads. Metrics can provide information on how the game is used and what parts receive more attention than others, but it is likely that forum interaction will still be used in the future to gain new ideas. If users suggest adding more of something that already exists in the game – for example, more variation to buses so their public transportation lines would look nicer – it can be possible to check the metrics to see how much public transportation is used generally by the players, to find out how much buses are used. If the buses are used by a large portion of the players, it is safe to assume they work well and make more of the similar buses. If they are not used by many people, different buses might be a better choice, for example buses with a larger capacity for carrying passengers or a higher average speed, to help make the buses a more interesting feature to play.

One further way of using metrics is providing information for tech support. If, for example, after filtering the results based on a certain graphics card or other hardware there are registered users but no data, it could mean people with said hardware cannot start the game. Also, reduced framerates can be filtered from metrics. Severe problems, like the game not launching or repeating crashes to desktop on starting a new game will be seen on the forums fast. For this I suspect the forums will be the best place to get the initial information that something is wrong, but to find out what is causing the problem, the metrics can help.

5.6 Confirming Design Choices

City-builders are a tricky game genre when it comes to measuring whether or not the game plays as intended. Since city-builders are sandbox games, they do not actually have a clear and definable wrong way of playing them. Exploiting a fault in the game or taking advantage of poor balancing between options are not things that absolutely break the game, they simply change it. While obvious exploits are typically weeded out in the testing phase, the finished game will most likely have more efficient and less efficient ways of playing it. However, since the player’s aim is not always to play as fast and efficiently as possible, slower progression is not always a sign that something is wrong. Sandbox games, like *The Sims* series (Electronic Arts, 2000), have a different player motivation than most other genres. While many games offer some means of self-expression, for example, character customization, sandboxes rely on the user’s creativity and the need to express themselves. This can be traced back to Maslow’s (1943) hierarchy of needs, which states that self-actualization is the highest need that only appears when other needs have been met. A sandbox is not about winning, it is about creating. The best sandbox game imaginable would support numerous different playing styles and thus the metrics it produced would be very varied. The best thing the metrics could show is that many different tactics allow for long play sessions, and that there are no sudden drops or peaks in the learning curve. The sandbox system should allow users to express themselves through playing, and have room for many play styles.

This does not work very well with metrics, because metrics rely on few key playing strategies to be recognizable and supported by the game. Using some sort of model to identify sandbox playing styles, in the same way as Bartle's model divides MMO players, would be really helpful when designing metrics.

One of the major challenges encountered during the development was the uncertainty whether or not the team is able to extract data from the servers and when. Using resources for something that might end up having no benefit at all is something a small team cannot afford.

The data gathered from the final released game does not represent the whole player population. Data is only gathered from users who register the game and allow data to be sent. An online game could have much better representation of players, since the game would be played online and all players' data can be documented. With a single player offline game, it is only possible to rely on players registering the game and allowing the occasional internet connection to send data to the servers.

As in many projects, time has been a challenge in this one too. The requirement from the publisher to have metrics surfaced in the middle of the project. Designing and programming could have been easier if it had been known from the start that metrics should be collected. Luckily, the game has many systems which collect information for the player (a statistics screen, unique buildings unlocking after player has finished certain tasks), so much of the information just needed to be sent to the telemetry server.

5.7 Implemented Metrics

In the end, not all of the planned metrics were implemented due to the time limit given for working on metrics. All of the required metrics were implemented, as were some of the additional ones. This was the first time working with game metrics for the whole team, so there might have been bugs in the formulas, meaning data was not counted correctly, which might then have resulted in incorrect metrics. Also, the inexperience might mean we might have made some larger scale beginners' mistakes and ended up with metrics that are not very usable in the end.

Table 3 shows the actual metrics saved on the server. One saved event can have up to six parameters saved at the same time. “Standard telemetry” below refers to the metrics required by the publisher. “Others” refer to the ones added by the developer.

Table 3. Events and parameters programmed to be saved into database

event	Description (When it's sent)	param1	param2	param3	param4	param5	param6
Standard Telemetry							
first_launch	First time the game is started						
start_game	Game is started						
exit_game	Game is terminated						
meta	Game is started	machineid (unique computer/device identifier)	machinemodel (Computer description)	gfxdevice (Graphics card model)	gfxversion (Renderer/Driver)	gfxmemory (Video memory)	

sysmemory (System memory)				guid		
cpucount (Amount of cores)				environment (name of the environment with new game)		gameTime the ingame time of the current map
cpu (CPU specs)				invert_traffic (is traffic inverted with new game)		simulationTime (time it took to load simulation data)
oslanguage (Operating system language)				map_name (name of the save)		scenes (time unity scenes took to load)
os (Installed operating system)				start_flag (NewGame, LoadGame, NewMap...)		main (main thread time it took to load)
gfxshadermodel (highest supported shading model)	optional: autologin			type (the Game, the Map Editor or the Asset Editor)	type	totalTime (time it took to load)
Game is started	User logs into the paradox account	User created new paradox account		Session loaded	Session ended	A loading finished
meta	login	create_account	Others	start_session	end_session	session_loaded

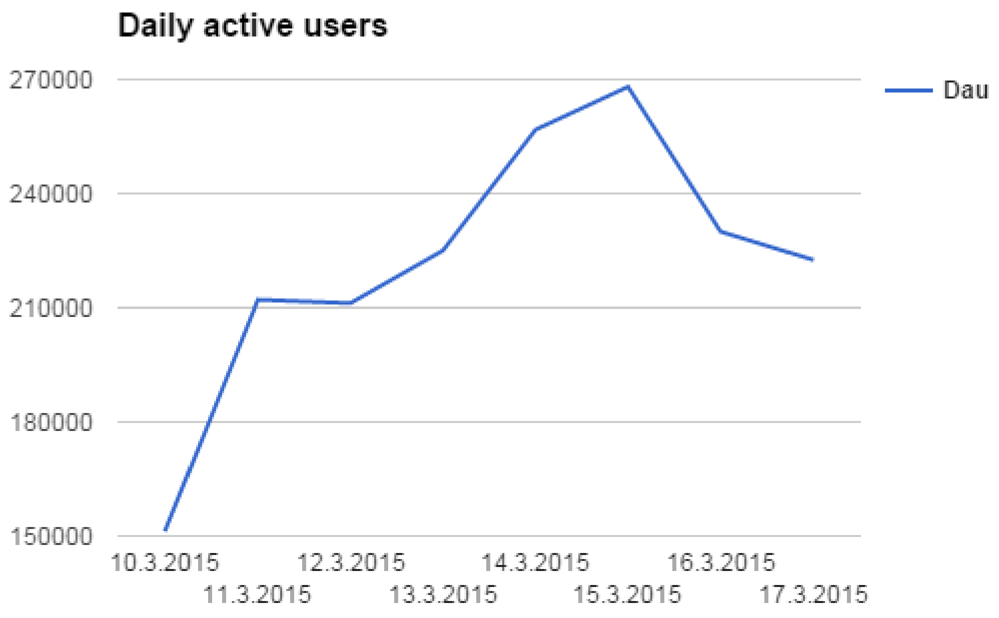
	modCount (total of installed mods)	message (human readable message if available)	gameTime		
buildings (number of custom buildings)	enabledModCount (amount of active mods)	type (internal type produced)	name		target (steam app id or url)
On custom content loaded	On custom content loaded	On intercepted error	Milestone reached	Main Menu Store button pressed	News feed link clicked
custom_content	custom_mods	error	unlocking	store_clicked	newsfeed_clicked

5.8 Analysing Metrics

After the game launched, everyone both at a development studio and at the publisher’s offices were so busy that metrics data did not get much attention. A month after launch there still had not been time or resources on the developer’s end to look into the metrics and what they could tell. The game was technically very solid at launch and had only a few bugs left, so no major bug hunting or fixing was needed. If something major would have been wrong, looking at the metrics would have been one way to pinpoint what the actual problem was.

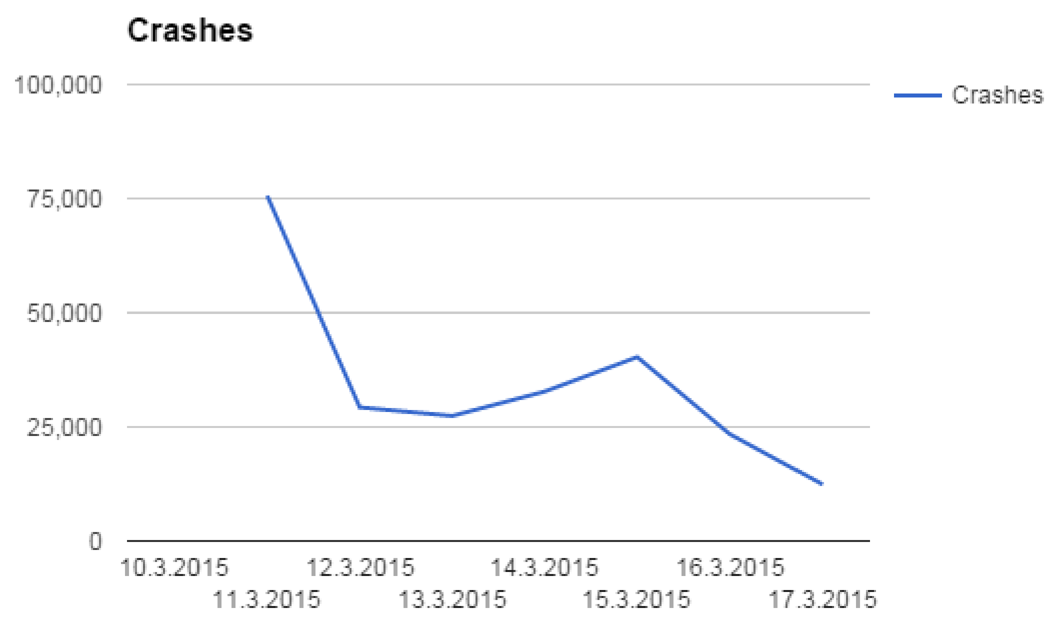
When the game was released, it soon became apparent that analysing metrics would be a hard task. Access to the database was unfortunately not available, and even if it had been, understanding and analysing the huge mass of data would have taken a very long time and required a powerful machine to do. With a small team, there simply was no one available to take the time to analyse the data thoroughly during the first month after the release. For this thesis, some data has been extracted by the publisher, but only selected data points and limited time scales, to keep the amount of data reasonable.

The information presented in this chapter is from the first week after the game was published. The time stamps used refer to Central European Time Zone (UTC+01:00) but the metrics are collected from players all over the world. The DAU numbers are based on individual Steam IDs. The numbers may contain some illegally downloaded copies but the validity of the metrics is not in any way affected by this.



Picture 5. Daily active users

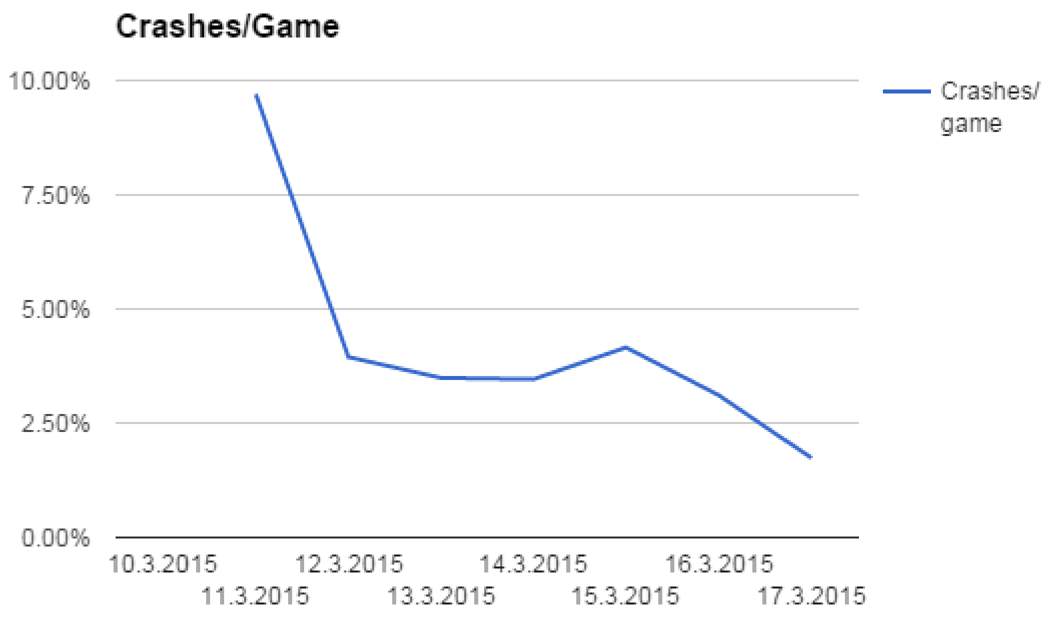
Picture 5 shows daily active users on the first week after the game was released. There is a very clear increase during the weekend (Fri 14.3, Sat 15.3. and Sun 16.3.), as people tend to have more time to play on weekends. The peak is on Sunday. All other graphs show the weekend peak in some way, for example the amount of crashes clearly rises as daily active user numbers rise. The daily active users graph in itself does not give much information, but helps with understanding the other graphs and their peaks.



Picture 6. Crashes

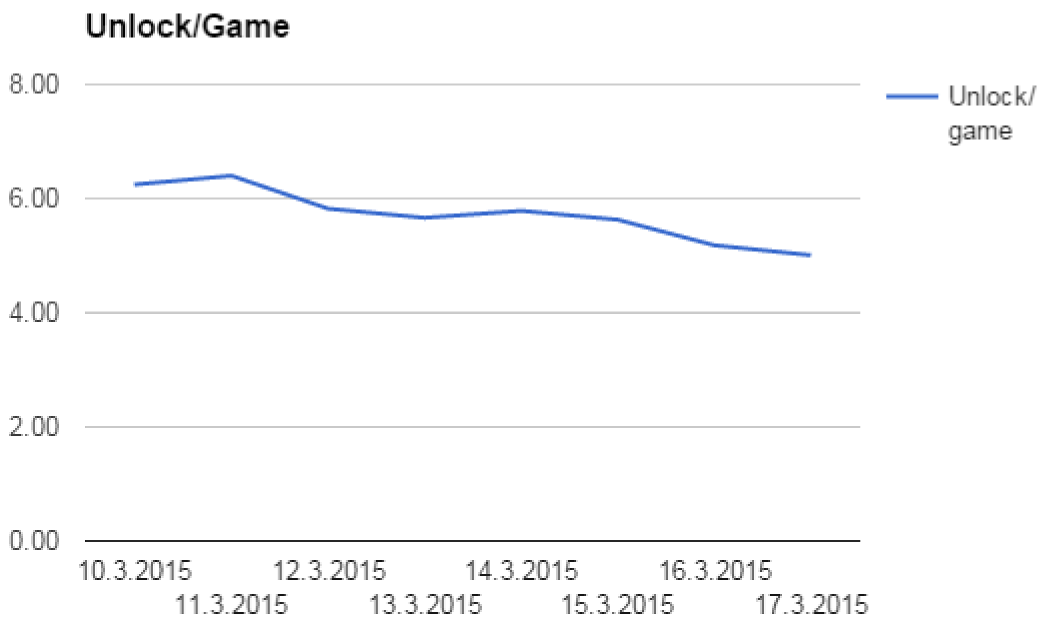
The graph in picture 6 compares the number of start-game events in the database to the exit-game events. Comparing these two numbers reveals the times the game did not shut down as it should have, but was forced to quit or crashed, not allowing the matching exit-game event to be saved. A game-start event is saved when a save is loaded or a new game started, marking the start of a game session. The exit-game event registers when the users exits the game state to the main menu or desktop through the exit menu of the game. Exiting with, for example, ALT + F4 does not record an exit-game event. The graph also shows that for some reason the telemetry did not work on the first day. All other data points were saved, but for an unknown reason the start and exit events did not log into the database. The sheet only shows us the actual number of game sessions that did not end in the game properly shutting down. The number of crashes drops hugely on the third day. There are two possible reasons for this. On the second day, a small patch was added to the game to help with some issues Mac users were having. There was also a bug that caused the game to often crash when exited, which was fixed too. This crash bug and the Mac problems are the most likely causes for the spike on the second day. Then again table 4 (on page 58) on operating system used shows that Mac operating systems make up for only 6.2% of all the machines the game is played on (note that this number is from a later date, but it is likely to give some indication of Mac user percentages) so the effect of the Mac problems might be very low.

We can also see a smaller spike on 15.3., which corresponds to the very peak of the daily active users on the first week. The number of crashes rises from the 13th onwards, which was a Friday, and when compared with Steam user information and the DAU graph of *Skylines*, is also the time when the number of users on Steam as a whole and especially on *Skylines* rises steeply. Steam publishes real time user numbers, which clearly show that people play the most on weekends, so a rise in the number of crashes correlates with the number of players. Once again, as this graph only shows the number of crashes, it does not tell of the likelihood of the game crashing.



Picture 7. Percentage of crashes

Picture 7 shows a graph that compares the number of game-started events to game-exit events to show a percentage of the start-game events that do not have a corresponding game-exit event, which means that the game was either shut down forcibly or crashed. While the graph before this shows the number of crashes, this one tells how the number relates to the number of game sessions. Like the previous graph, this one is also missing the data from the first day. We can still see a spike that raises the percentage of started sessions not shutting down properly go all the way to almost 10%. This number seems high and is most likely due to the crash on exit bug mentioned previously. The bug being fixed on the 12th more than halves the percentage of sessions crashing, telling us that the fix seems to have helped a huge number of the people suffering from the problem. The smaller spike from the graph showing the plain numbers of crashes is replicated in this graph. The reason for the spike may still be different, because the rest of the graph does not replicate the number of crashes graph. While the number of crashes rose from the 13th to the 14th, the percentage of crashes slightly dropped. From the currently available data, no reason for the spike on the 15th can be found.



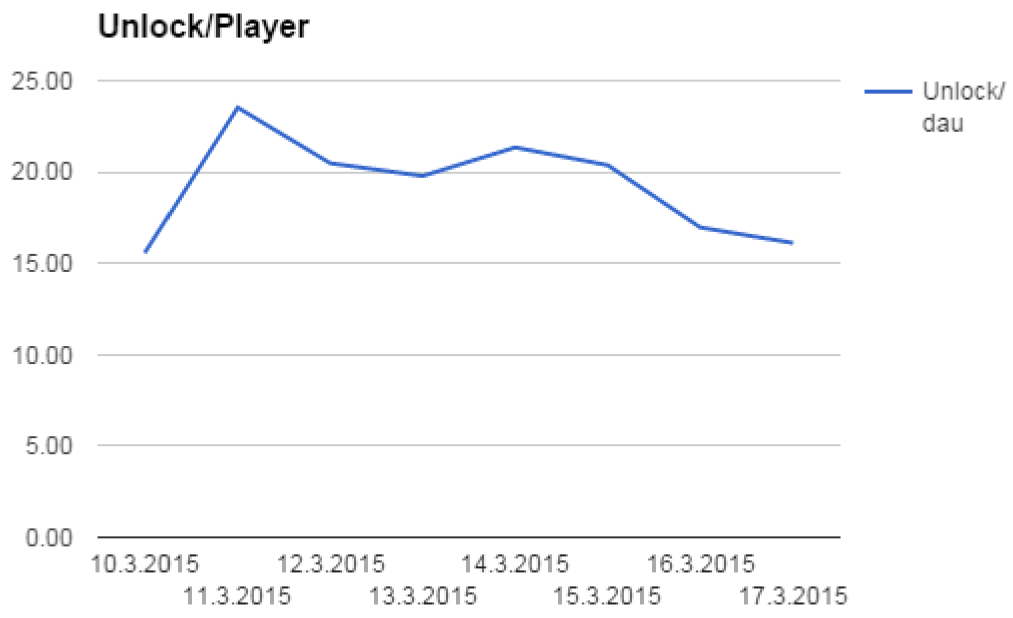
Picture 8. Unlock events compared to game-start events

Picture 8 shows how many unlock event have happened per game started event. This graph tells how much content a player unlocks in a game session, as each time a save is loaded or a new game started, a game-start event is saved in the metrics. One unlock refers to one in-game milestone reached. The game has 13 milestones in total and was designed so that reaching the last milestone should take about 7 hours of playing, depending on player’s skill level and playing pace. The milestones are tied to population. The number of population needed for next milestone rises steeply as the game progresses. It is not possible for the player to alter the order of the milestones so it is safe to say that, on average, players had progressed to milestone six by 10.3. Milestone six requires the player to have 7000 citizens living in the city. To give an idea of the curve the population requirements are on, the last milestone, number 13, requires a population of 80 000 people.

The main reason unlocking was decided to be tracked was to try and see if the pacing for it is right. Clearly players do reach milestones (see page 23 for table of milestones and their requirements) and based on the average number of milestone unlocked per session, they seem to be prefer game sessions that last a few hours at a time. However, the pace at which a player unlocks milestones can vary greatly. Based on the testing done during development, players have different styles of playing and their skill level and familiarity with the genre affects the pace of the game greatly. Very skilled players who are familiar with the genre and especially *SimCity* games can play very fast and unlock milestones at a rapid pace. They can however actively choose to play slowly, and some players like to do other things while they play, such as watch movies or pause the game to browse the web for some time. The game supports differently paced playing styles, so these are not a right and a wrong way to play the game. No testing was done prior to launch as to how long it takes to reach milestones, so there is no data to compare the number of milestones unlocked to, so only guesses can be made of what a little above six milestones per session would mean time-wise.

What the graph does tell is that the number of unlocked milestones per sessions seems to hover between five and six. If further testing is done or other metrics implemented to find out the average length of a play session, the milestones could be optimized to work for the sessions, or perhaps the one the players usually stop at could have some extra incentive to continue playing. There could be something in milestone five that makes the game cumbersome to play, or just that milestones six and seven have too few things unlocking and feel boring. It could just be that unlocking six milestones is what can be unlocked with the most popular playing style and pace in the most popular sessions length, whatever it may be.

The graph has a small spike on the second day, then fluctuates for three days and begins to decline. If players continue playing the same saved games, the number of unlocks will go down as progress slows the further the player gets. Eventually unlocks stops altogether, as the last milestone is reached. Reaching the last milestone in no way means that the playing stops or that the game has reached an end, the player can continue to work on the city as long as they like. It is possible that day one is the best choice to look at if looking for data on first play session and how many unlocks happen in it, but because of time zones and the structure of the game, players can play many short new games, which raises the number of milestones unlocked on average, and because not all players start at the same time, the data cannot be thought to represent simultaneous play sessions started at the same time.



Picture 9. Unlocked milestones per daily active users

While picture 9 does not show unlocked milestones per individual player, it tells how many unlocks happen per daily active user. While there are clear spikes in the other graphs on the 15th, which is a Friday and thus a day when many people have more time to play than on other days, the unlocks per player graph does not show a spike there. This can mean that the spike in the other graphs shows that player numbers rise, but these players have maybe played on the previous days at different times to unlock the first and thus fastest milestones, and then return to the game on Friday when they can play for longer. Comparing the DAU and unlocks does not seem to give much usable information, as it does not tell if the unlocks happen during the same session. When compared with picture 3 that shows the graph of unlocks per session, it seems that many users play many sessions during one day. The unlocks per session graph shows that on average 5-6 milestones are unlocked per play sessions, meaning the time between a start-game event saved on loading a save or starting a new game, and an exit-game event that is saved when the user exits the game state to main menu or desktop. When comparing these 5-6 unlocks on average per sessions to the total number of unlocks, it seems that players play approximately 3-4 sessions per day. This data does not reveal the length of these sessions, or if they are new games or previous saves.

Table 4. Operating systems used by players (data extracted on 24.4.2015)

Operating system	**Percentage of games played**
Microsoft Windows	92.2
OSX (Mac)	6.2
Linux	1.2
Other/Unknown	0.4

The use of different operating systems shown in table 4 tells what operating systems the game is played on. The database lists each individual computer the game is played on. Since a single Steam user can have several devices but still use the same account on all of them, the number of computers played on is higher than the number of individual users. While these numbers are from a later date than the data the graphs are based on, it is still quite safe to assume that the percentages have not changed from the time when the graphs were created. Windows is altogether so overpowering that most of the data will most likely come from systems operating with Windows anyway. Still, Mac and Linux users are very vocal on the support forum and have reported quite a few problems they have run into. For example, searching for "Linux" brings up nine pages of content on the Support & Bug Reports subforum on Paradox Plaza *Cities: Skylines* forum, when in total there are 63 pages of discussions. This is 14% of all the discussions which is a lot when only 1.2% of all games are played on these operating system. Based on just forum feedback, it would seem that a large number of players are affected by Linux bugs, when the actual percentage of games played on Linux is very low. Looking at the operating system table, it is clear that even if all games on Mac and Linux operating systems would have the game crashing constantly, they would at most still make up only 7.4% of all the individual computers played on. Naturally, those who have problems are much more vocal in the Support forum, as it exists solely so that users can submit problems and get answers or fixes. Simply by looking at the forum feedback it is very hard if not impossible to try to find out how many players are affected by problems, and this is where metrics come in to help us get a better picture of how many users are affected and if it is possible that all users on a particular operating system are affected.

5.9 Summary

Table 5 is a summary of the findings of this thesis. The observations made during the design work, implementation and analysis are listed on the table divided by the phase of the work the finding is relevant to.

Table 5. A summary of observations at each phase of designing metrics, analysing data and iterating based on it.

Phase	**Observation**
Preparing for designing metrics	• Choose carefully what information would be beneficial to iterate the game later. • Clear questions that the data should answer work well. • Keep the number of metrics as low as possible to avoid huge amounts of data to analyse later. • For a game that is inspired by tried and true classics, the main parts that need metrics to prove they work are the ones that do not follow the classics. Focus on those. • Try to anticipate the resources that will be available for analysing the data. If it is unlikely that there will be available personnel dedicated to analysis only, keep the number of metrics even lower. • Some sort of metrics should be designed to map how much players use each feature of the game.
Designing metrics	• Professionals of all aspects of game development are needed, but a designer who knows the game system well and can formulate questions for the data, and a programmer who knows what data is already being collected so that as large amount as possible of the work already done can be utilized are especially important. • If metrics are designed at the start of the project, the other aspects of the game can be built around them. If metrics design happens in the middle of a project, using data that the game already collects (achievements, player actions to unlock features, graphs shown in the game etc.) is essential to save time and resources.
Analysing data	• Analysing is made much easier if the amount of data is not huge. The metrics needed should be kept to a minimum, as is noted on the preparing phase above. • Try to find correlating graphs and pay attention to points in time when there has been a noticeable change. For example: what happens to the number of crashes after the game is updated • Spikes and drops are what should be looked for. • It is possible to either take a look at the data and then try to find answers for sudden changes in it, or to choose an event and then look at its effect on the data
Iterating on the game or creating more content	• Metrics should not be used on their own, but together with feedback from users and critics. • Often, iteration means adding new metrics to track based on feedback received through other channels. • Some issues cannot be researched with metrics. For example: players having problems with laying roads does not produce any reliable metric, as both the confused and the non-confused players place and delete many roads.

	• Metrics do not provide information on how meaningful the game is to the player. • Metrics for measuring how much each feature is used (as mentioned in the preparing phase) can be utilized to identify the least used features so they can be made into more integral parts of the game, or to identify the most used features and offer players more of the same.

6 DISCUSSION

When comparing this thesis to earlier published research, some similarities can be found. While most of the texts on game analysis, metrics and telemetry concern mobile or social games, or games with a different monetization model than *Skylines*, some earlier texts can easily be applied. The graphs generated from *Skylines*' telemetry data can be compared to each other and used to help in understanding what causes peaks and drops. Hazan (2013) has a collection of case studies from well-known studios on games that are not on mobile platforms. Below, I compare these cases to *Skylines*' case and see what is done similarly, what is different and what cannot be done based on the data available.

One case study looks at how the possible causes for *Splinter Cell*'s (Ubisoft, 2002) player drop off rates were investigated by comparing different graphs and looking for correspondences. While *Skylines*' graphs clearly show even to the untrained eye for example how the actual number of crashes rises with high daily active user numbers, but percentage of games crashing stays almost the same. *Splinter Cell*'s analysts compared average saves per level to players dropping the game after a level to see if the game was too hard, and saw that this was not the case, the graphs did not show similar patterns. Graphs with similar patterns show that the data they visualize is most likely related in some way, but when graphs do not show similarities, the data is not related.

In a case study of camera use (Hazan, 2013) in *Prince of Persia (Wii)* (Ubisoft, 2010), the aim was to track if camera controls were too hard for the players to use. The initial data on the number of times the camera was used and the locations where on the levels the camera was used proved that, yes, players did use the camera controls. The fact that the camera was used still did not reveal if the users had problems with it, much like how in the *Skylines*' graph about unlocking milestones during play it can be known that players do unlock milestones, but if the intervals are good or if the players are having trouble reaching the milestones remains a question. With *Prince of Persia*, the reason to look into camera controls was born from the concern that they might be hard to use for players. To find out more, *Ubisoft* used questionnaires for play testers with a few questions specifically on how the camera controls feel. This revealed that players did not feel the controls were hard to use and that they were often unaware of how often they had even used the camera controls. In *Skylines*' case, dedicated testing of one area of the game is not currently needed, because the player feedback on the forums shows no clear problems. So far no threads on milestone pacing have come up, so they seem to not bother players. If *Skylines* would suffer from a problem like the expected camera control difficulties that were investigated with *Prince of Persia*, they would show up on the forums.

The analysis on *Crackdown* was referred to earlier in the thesis (see page 8). While *Crackdown* is an excellent example of how metrics can be used to confirm design choices and to find out what it is that gives players the feeling the design aims for, the analysis on it cannot be easily applied to *Skylines*. There is currently no data on the fun factor of *Skylines* relating to single actions taken by the players. To find out if certain actions increase the feeling of fun, like the roof-top jumping in *Crackdown*, there would need to be quantitative data on what players do and data on how fun these players rate the game. It is very possible that some playing styles in *Skylines* may produce a game experience that feels more fun than others, but with current data, it is impossible to find out.

The case of *Halo* (Microsoft, 2007) relates to a situation where players feel frustrated and the reasons why. A questionnaire was used to ask players what they were feeling and the answer and location were marked on maps to show where in the maps players found the game frustrating. This combination helped developers see exactly where the players needed more guidance or a different level of challenge. As *Skylines* does not have a player avatar, using locations is trickier. It could still be possible to do by tracking the cursor movements on the map. The system of using a questionnaire that pops up every three minutes feels like something that might affect player experience in a negative way, because interrupting the game usually feels frustrating. With the current data, *Skylines* does not have any location based metrics.

Texts concerning mobile games and the use of metrics are very different from this thesis, because of the different monetization model *Skylines* has. While free-to-play games and social games use metrics to help design games that maximize monetization and keep players playing, *Skylines* uses the classic monetization model of users paying to get the game and then being able to play it any way they want as long as they want with no additional payments. Analysing data is similar, but the aim is different. When

Bilas (2004) talks about different types of analyses and shows graphs of what boosts players use most compared to the level they are playing, the aim of the graph is to find out if the boosts cost the right amount in relation to their effectiveness. In *Skylines*, the Monuments and their effects could be compared, but the reason would not be to find the right cost for them, but to make sure all have an equally powerful effect on the play and none feel too overpowering or weak when compared to others.

7 CONCLUSIONS

Like Drachen et al. (2013) suggested, choosing beforehand the exact questions you need answers for seems to work very well. In addition, the development team wrote guidelines to help further narrow down what questions should be asked. This helped a lot when working with a very tight timeframe. The suggested metrics for simulation games in the article were very sparse, but the general idea worked well for *Skylines*. Player actions are the interesting thing, and to design metrics for a specific game, looking at the actions and narrowing down which actions can produce meaningful data is essential to avoid having a huge amount of data with a lot of noise, which makes analysing difficult.

While metrics are useful, and with proper analysing can help identify problems in the game, they cannot replace all tools used for getting feedback from players. Polls, general discussions on forums, threads asking for ideas, making players a part of the development process by letting them suggest changes; all of these things need ways of collecting data other than metrics. Metrics also completely lack the benefit of making the players feels like they are a part of the development process; that they are heard and their opinions are respected. While metrics do allow players to participate in the development with their game play choices, they get no feedback that would tell them that someone is actually listening and interested in their choices. Based on Colossal Order’s experiences with communicating with the player community, presence on forums will definitely continue alongside collecting metrics. If possible, information gained from the forums will be used to explain metrics further. Metrics can also help in understanding what people on the forums are discussing, for example what they refer to as a game session may be close to the average game session play time displayed by the metrics. If some topics on the forum seem interesting but need more data to be fully understood, new metrics could be designed to help gain understanding of what players do and how they feel.

7.1 Benefits of Using Telemetrics with Game Design

Metrics are useful in finding out if and how much a feature is used among players. This information can then be used to guide the design of expansions. There are two ways to use the information on how much a feature is used. The most used features are popular, and creating more content like them is very likely to sell well. The other way would be to try to give attention to the less used features to get them to become a more important part of the game. Simple statistics of how much a feature is used can be beneficial to design and help guide the design process away from pure intuition and towards more fact based decisions. Designing metrics can help the designer focus on the main player types and playing styles that the game should support. Designing metrics without this knowledge is impossible. Who is the game for? How should they play it? What types of playing styles are supported? Is it possible to leave room for other styles as well?

Sandbox games seem especially challenging when it comes to metrics. With more straightforward games genres, like shooters or strategy games that have clear, simple player actions that are expected and can be anticipated, measured and encouraged, designing metrics is relatively easy. Sandbox games rely on the player’s creativity and work towards fulfilling the need for self-expression (Juul, 2002). There are also different types of players who enjoy different ways to play a game (Bartle, 1996, Kallio et al. 2011). This means that many different playing styles need to be supported to allow for different players to express themselves in a way they feel is fulfilling. When a shooter game is more about learning how the system works so that obstacles and challenges can be overcome, a sandbox offers the user interlocked systems that react to player actions and tools to express themselves through the game system. The more playing styles are supported, the more varied the metrics seem to become and the amount of useful data grows.

Metrics can tell how players use the game, not why they do what they do. In simulation games, there is no clear right or wrong way to play, and metrics are very hard to read if the goal is to find out if the game creates meaningful play. "Meaningful play in a game emerges from the relationship between player action and system outcome; it is the process by which a player takes action within the designed system of a game and the system responds to the action. The meaning of an action in a game resides in the relationship between action and outcome" (Salen & Zimmerman, 2003)

This concept of meaningful play does not show in metrics. Based on forum discussions about traffic, handling traffic seems too hard for some players. The following is a prime example posted on the *Skylines* forum on Paradox Plaza, the publisher’s portal for all information on their games by user K-4U: “I have the same problem. 6 lane one way road going around my industrial area.. (...) They would stick either on the far left lane, or the far right lane. The lanes inbetween (sic) were not used at all!” A number of players seem to be having trouble with traffic, feel that it is bugged and come to discuss it on the forums.

Those who start to understand how traffic works seem to find it interesting and make meaningful choices in traffic design. A reply made by user Darkath to K-4U’s comment: “Your traffic would be clogged anyway here because there are far too much traffic lights/intersections in a small space ...”. This comment shows that Darkath understands that the vehicles queuing on one lane is not a bug, but an indication that the player needs to tweak their traffic system. User IVIaarten tunes in: “I can't comment on CiM2, but in this game you can fix it yourself by learning how the traffic pathfinds (sic), and making sure they actually use the turning lanes.” The user refers to traffic problems in a previous game, *Cities in Motion 2* (Colossal Order, 2013), which some other users in the discussion felt had carried on to *Skylines*. These types of discussions happen often when some users ask about traffic problems or possible bugs in it, and others are of the opinion that traffic is part of the game and users need to learn how to control it. The latter is the design intent of the traffic: to offer meaningful play and some challenge.

Some tutoring or guidance is needed in addition to what there is now, because all players do not find the meaningful play in traffic. This is a thing that is very hard to see in metrics. The only thing that might show that traffic is not meaningful play for a large portion of the audience could be the number of roads placed and upgraded. Still, the number alone is not very telling, as players who find traffic hard to grasp try to solve the traffic problems by placing many roads. Those who do think of traffic as meaningful play also place many roads, but have a better idea of what they are doing, and thus feel their actions to be more meaningful.

Many things in the design seem to have gone just right judging from player feedback and reviews by critics. The main things are hard to confirm, as many systems in the game aim to support less skilled players and teach them how to play. These systems create the experience of having the city on the brink of catastrophe but the players being able to save it at the very last minute. This is likely to strengthen the bond the players have with their city. Actually the game goes into a loop, telling the player that there is a need or something essential is missing. The loop goes on for a long time, giving the player time to react and to learn what is wrong and how it should be fixed. There is no game over state and the city recovers well from setbacks, so the player can and will have experiences of saving the city, of learning how to play better, of mastering the game. This creates positive emotions associated with the game and still gives a feeling of challenge, of the game not being too easy.

In this case, confirming design intuition seems to be better done by gathering feedback from forums than looking at the metrics, but this may be due to inexperienced metrics designers or the short time spent on the metrics design. Literature on metrics does not touch the concept of meaningful play so this might be a new area to explore further in future research. Is meaningfulness even possible to track with metrics? If it is, what sort of metrics can measure meaningfulness and how should the analysis should be? How can frustrated playing and meaningful, motivated playing be identified in metrics?

7.2 Specifics of City-Builder Games and Use of Game Analytics

Just like Drachen et al. (2013) note, simulation games have very varying metrics. Finding the ones that are useful for a specific simulation game requires looking into possible player actions and choosing key questions that need answers. Due to the sheer number of playing styles simulation supports, it would be impractical to track all actions. The amount of data would be huge and analysing it would take an immense amount of time. There are no ways to play the game “wrong”, as the core of the game is to play with the possibilities of the world and to create things. Bartle's (1996) player model would place simulation players in the explorer or achiever category or both, so looking into metrics done for other games that have an emphasis on exploration or achieving could be helpful. Social games use metrics to guide design and monetize, and thus most of the games using metrics do not have much to offer for simulation games that use a different monetization model.

Stickiness, the number of returning players, is interesting for all games. If there are online features, registrations or other ways of connecting users to the net and marking them as individuals, stickiness can be tracked. These metrics still only speak for the popularity of the game and how often players return, not why they want to play the game, what they do in the game or what makes them return.

Metrics tell a little bit of what players do in the game and when, but it is very hard to recognize motivations and playing styles from the mass of data. More data points might have helped, but it seems that designing metrics for simulation sandbox games is among the hardest of genres to get useful information out of. Players do the same things if they feel a feature is meaningful or if they feel it is not and they are just trying to get it to work. Finding evidence of meaningful play in metrics is very hard. For this purpose reviews, YouTube play videos and forum discussions seem like better ways of finding information.

Based on the work done with this thesis, implementing telemetry for simulation games is no small task and should be taken on with care. Plenty of time and expertise is needed, and literature on the subject is sparse.

Specialized theories on how sandbox games are played, what type of players play sandbox games and how they play them would have helped a lot in designing the metrics. In the case covered in this thesis, the design was mostly based on intuition because of the time and resource limitations. A model like Bartle's (1996) but for sandbox players would be ideal.

One question to ponder on is do sandbox games always benefit from supporting as many playing styles as possible or if there is a sweet spot where a certain number of styles being supported is enough? Sandbox games are a unique game type that along with goals and learning promotes self-expression through playing. If supporting the most styles is what benefits the game most, metrics are very hard to design, because collecting data to represent each playing style would result in huge amounts of data that would be fairly fragmented as all styles are most likely not equally popular.

7.3 Limitations

The main limitations for designing metrics were time and the inexperience of the team designing the metrics. This study would have benefited from more time spent on designing the metrics and more actual metrics data being gained. Having no direct access to the collected data meant that the data analysed was a very small sample and could only give a small overview of how the launch week went, but most of the data has not been analysed at all and the amount of data is huge.

The number of written sources on game analytics is quite low. The practice is fairly new and has not been studied much, and it is likely that much of the knowledge on the subject is held by companies that offer analytics services. Most of the sources concentrate on social games that are very different from a single player city-builder and thus much of the written material cannot be applied to this case.

Still, this case study offers a good overview into the process of designing metrics, applying them and doing light analysis on the data. Future applications of the data and possible iterations are also discussed.

7.4 Future Expectations for Telemetrics in Cities: Skylines

When more playing hours are recorded, the data will most likely start to lose spikes and drops. Conversely, at launch the information is not reliable due to the small sample size. For designing expansions, telemetry data will be used to find out what players particularly enjoy in the game. This data will be used alongside forum comments and critical reviews, so the information that is used as a base for decision making would be varied and represent the community well. When compared to earlier games, this inclusion of telemetry data should help gain information on the silent users, those who are not active on forums or other channels. As Nilsson and Södegren (2014) found, when all users have the possibility to be vocal, it is often hard to actually hear what everyone is saying. Some users are active and discuss their opinions on the forums, others are not interested in talking about the game or belonging to an online community and thus do not share their opinions. When using telemetry as additional data alongside forums, the active forum users who are vocal and have strong opinions will still be heard as before. The telemetry collects data of all users, so those who do not actively promote their agenda on forums are somehow represented in the information that future expansions are based on.

If needed, more metrics can be added with expansions. So far none have been planned, but there has not been much time to even look at the current data. New metrics would not be included in expansions that cost money, but rather in free updates so as to make sure as many users as possible get the new version with the new metrics. Expansions are not purchased by all users, so unless something specific to an expansion needs to be tracked, the free updates are better suited for new metrics.

A closer look at different playing styles would be interesting and possibly provide some data that could be used to make the game better. Based on forum discussions, some players feel the traffic system is not working properly. This is not actually the case, but optimizing traffic is the hardest part of the game, and requires understanding how the system works. Others find traffic interesting and feel it is a meaningful part of the playing experience. The current assumption by the developer is that the traffic logic is not explained clearly enough to the user in the game, which makes players confused and results in some of them being unable to understand how traffic works. If the player does not see the traffic as logical or meaningful, they dismiss it as bugged and do not gain the enjoyment of playing with the traffic management possibilities that the game can offer. Naturally, some players just do not enjoy traffic management, but that would not make them feel it is bugged. Based on my intuition, players are usually right when they say something is wrong with a game system. They seem to be adept at identifying the feature that is not working like they expect. Often their suggested solutions are far from ideal and at most times are based on the assumption that simulation games need to be realistic. This leads to them thinking that if something in a simulation game feels “off”, it must be because it is not realistic. The suggestion on how to fix the issue is then to make it more realistic.

Traffic issues seem very hard to track via metrics, but some further work will be put into trying to find what the problem is and if telemetry could be used to find out how to fix it. It is possible that with the traffic system the only realistic possibility is to gather forum feedback and do more testing to find out what sort of guide messages or tutoring best helps players find the meaningful play in traffic, but using telemetry to support making the right decisions would be highly desirable.

REFERENCES

Research literature

Bethke, Erik (2003). *Game development and production*. USA: Wordware Publishing, Inc.

Crawford, C. (2003) *Chris Crawford on Game Design*. USA: New Riders.

Davenport, T. H., & Harris, J. G. (2007). *Competing on analytics: The new science of winning.* USA: Harvard Business School Press.

Drachen, A. et al. (2013). Game Analytics - The Basics. In Seif El-Nasr, M., Drachen, A. and Canossa, A. (eds.) (2013) *Game Analytics - Maximizing the Value of Player Data*. USA, Springer.

Gee. J. P. (2005). *Learning by Design: good video games as learning machines*. *E-Learning, Volume 2* (Number 1).

Hazan, E. (2013) Contextualizing Data. In Seif El-Nasr, M., Drachen, A. and Canossa, A. (eds.) (2013) *Game Analytics - Maximizing the Value of Player Data*. USA, Springer.

Juul, J. (2002). The Open and the Closed: Game of emergence and games of progression. In Mäyrä, F. (ed.) *Computer Games and Digital Cultures Conference Proceedings*. Tampere: Tampere University Press.

Kallio, K., Kaipanen, K. and Mäyrä, F. (2011) At least nine ways to play: Approaching player mentalities. *Games & Culture*, 6:4, pp.327–353

Kultima, A. (2014). Pelinkehittämisen periaatteita. In Krokfors, L., Kangas, M. & Kopisto, K. (eds.) (2014). *Oppiminen pelissä. Pelit, pelillisyys ja leikillisyys opetuksessa*. Tampere: Vastapaino Oy.

Maslow, A.H. (1943). A theory of human motivation. *Psychological Review* 50 (4)

Mäyrä, F. (2008) *An Introduction to Game Studies*. USA: SAGE Publications.

Nilsson, M., Södergren, P. (2014) *Social media use in digital product development*. BSc thesis. Umeå University.

Paavilainen Janne, Hamari Juho, Stenros Jaakko, Kinnunen Jani. (2013). Social Network Games: Players' Perspectives. *Simulation & Gaming* 44 (6), 794-820.

Rollings, A. & Adams, E. 2003. *Andrew Rollings and Ernest Adams on Game Design*. USA: New Riders Group.

Salen, K. & Zimmerman, E. (2003). *Rules of Play*. USA: MIT Press

Games

Bioware. (2009-2014). *Dragon Age –series.*

Blizzard Entertainment. (2012). *Diablo 3.*

Colossal Order. (2013). *Cities in Motion 2.*

Colossal Order. (2015). *Cities: Skylines.*

Electronic Arts. (2013). *SimCity.*

Electronic Arts. (2000). *The Sims.*

Maxis. (1989). *SimCity.*

Maxis. (2003). *SimCity 4.*

Playfish. (2011). *The Sims Social.*

Sierra Entertainment. (1992). *Caesar*.

Sierra Entertainment. (1999) *Pharaoh.*

Sierra Entertainment. (2000) *Zeus.*

Wooga. (2010). *Happy Hospital.*

Wooga. (2011). *Magic Land.*

Zynga. (2009). *Farmville.*

Zynga. (2009). *Mafia Wars.*

Blog entries and Internet articles

Bartle, R. (1996) “Hearts, Clubs, Diamonds, Spades: Players Who Suit Muds”. Retrieved from http://mud.co.uk/richard/hcds.htm

Bilas, A. 24.11.2014. “Data Scientists Need Not Apply: How Anyone Can Do Game Analytics.” Retrieved from https://www.youtube.com/watch?v=qyBz7TyHzak. Watched on 26.4.2015.

Cities: Skylines Wiki. 2015. Retrieved from http://www.skylineswiki.com/Cities:_Skylines_Wiki.

Diablo Somepage. 2015. Retrieved from http://diablo.somepage.com/

Drachen, A. 27.7.2012. “What are game metrics?”. GameAnalytics Blog. Retrieved from http://blog.gameanalytics.com/blog/what-are-game-metrics.html

Cheshire, T. 5.1.2012. “Test. Test. Test: How Wooga Turned Games Business into a Science”. Retrieved from http://www.wired.co.uk/magazine/archive/2012/01/features/test-test-test

Elliot, P. 19.2.2010. “Metrics: The Key to Successful Social Game Design - Zynga”. Retrieved from http://www.gamesindustry.biz/articles/metrics-the-key-to-successful-social-game-design-zynga

Glajch, S., Shea, A., Tyrrell Jr., J. (2006). “Game Tutorial Analysis”. Worcester Polytechnic Institute. Retrieved from https://www.wpi.edu/Pubs/E-project/Available/E-project-030206-173827/unrestricted/Game_Tutorial_Analysis.pdf

Kennerly, D. 15.8.2003. “Better Game Design Through Data Mining”. Retrieved from http://www.gamasutra.com/view/feature/131225/better_game_design_through_data_.php

Paradox Interactive. 14.2.2015. *Infographic on Crusader Kings 2*. Retrieved from https://www.facebook.com/ParadoxInteractive/photos/a.430942739574.223458.166743484574/10153158992809575/?type=1&permPage=1

Sinclair, B. 15.4.2013. “EA shutting down Playfish games”, Gamesindustry.biz. http://www.gamesindustry.biz/articles/2013-04-15-ea-shutting-down-playfish-games

Techopedia. “Free to play”. Retrieved from http://www.techopedia.com/definition/27039/free-to-play-f2p

Valadares, J. 16.8.2011. “Measure or Die: Analytics and F2P Mobile Game Design”. Retrieved from https://www.youtube.com/watch?v=J0mWGDZO8QI

Wikipedia. 2015. “The Sims Social”. Retrieved from http://en.wikipedia.org/wiki/The_Sims_Social

Wikipedia. 2015. “Video Game Publisher”. Retrieved from http://en.wikipedia.org/wiki/Video_game_publisher

Wikipedia. “Wooga” Retrieved from http://en.wikipedia.org/wiki/Wooga

W3.org 31.8.2001. “Micropayments overview”. Retrieved from http://www.w3.org/ECommerce/Micropayments/

Paradox Plaza forum for Cities: Skylines, 2015. Retrieved from https://forum.paradoxplaza.com/forum/index.php?forums/cities-skylines.859