1 Introduction

1.1 Purpose of the Study

The main purpose of this thesis is to incorporate the two research areas Microservices and Online Games. Both areas have individually been the topic of intense research lately TODO: Citation needed but no research exists on how to combine the two.

In regard to the Microservices domain this thesis serves as a base for discussion about Microservice applications. Specifically the development and operation a set of Microservices is explained in detail. Of particular interest is the topic of how to facilitate the composition of Microservices and how to deploy a Microservice application in a modern could environment.

This thesis also provides a proof of concept that it is possible to develop Online Games using Microservices. For this purpose a vertical slice prototype is developed to showcase the necessary fragments that make up a working Online Game in the Microservice world. Furthermore the prototype serves as a simple development example of how to reproduce the development process of an Online Game. Is intended to live on after this thesis ends and will continuously improved in the future.

As a secondary objective the usability of the recently very popular TODO: Citation needed Procedural Content Generation (PCG) is also discussed in this thesis.

1.2 Theoretical framework

To curtail the infinite amount of research problems that the topics Online Games and Microservices offer a theoretical framework can be used. Within the framework the abstract research problems are translated to research questions that cover the area of interest sufficiently. According to the research questions hypotheses can be formulated. The verification of these hypotheses is the final goal of this thesis.

1.2.1 Research Problems and Research Questions

Usability of a Microservice for Online Games

Research Problem: The usability of Microservice suited to develop and operate an Online Game.

Research Questions:

- Can an Online Game run in a pure Microservice environment?

- Is a Microservice influenced architecture suitable to design an Online Game?
- Is the added complexity during development tolerable?
- Is the overhead introduced by Microservices negligible?

Deployment and Operation

Research Problem: The deployment and operation of a Microservice application in a cloud cluster-environment.

Research Questions:

- Which steps are necessary to operate an Online Game in a cluster-environment?
- How well do Online Games operate in a cluster-environment?
- How does a modern container engine assist in building and operating a Microservice driven application?
- How can the deployment process be automated?
- Which tools exist that help the developer with deployment?

Microservice Composition

Research Problem: The composition of a set of Microservices to form a coherent distributed application.

Research Questions:

- What degree of composition is necessary for a Microservice application to work?
- What is the minimal set of static and dynamic information that is needed for Microservice composition?
- Which of the common paradigms orchestration and choreography is suited better for Microservice composition?
- Which middle-ware is most useful in providing Microservice composition?

Microservice Coupling

Research Problem: The degree to which Microservices are semantically dependent on each other.

Research Questions:

- How can multiple game features be developed in parallel in different Microservices while preserving the overall application behaviour?
- What problems are introduced by coupling game features loosely?
- How does the consistency vs. performance trade-off impact Online Games?
- What protocols help to reduce service coupling?
- Can global static domain information help to reduce Microservice coupling?

Development of a Microservice application:

- Usability

1.2.2 Hypotheses

To be more specific about the validation it must be defined what knowledge is elaborated in this thesis. This is described by asking the major knowledge questions.

1.3 Significance of the Study

Today we we see a large number of game development companies. There is a large number of companies that have established themselves over the last year TODO: Citation Needed.. These triple A companies (AAA) have the budget to assign hundreds of workers to game projects with a budged of millions of dollars TODO: Cite to GTA5.

In the contrary we have many small independent that work in small teams. The keep the right on the intellectual property of their games and are not restricted by any financial backers. These companies need other types of financing.TODO: Backup with facts. These factors limit the companies in the extent they can do games in.

Hence indy-teams don't have the manpower to develop rich online game experiences from scratch because they lack the manpower. The fact that AAA companies don't share their knowledge about online game development further complicates this problem. The lack of tools in this area is another reason that indies struggle with dense online games. On the needed level of detail simply no middle ware exists.

1.4 Definitions

1.5 Terminology Definitions

1.5.1 Simple Online Game

A Online Game where where players are responsible to set up game servers. In these environments the players make the game rules. Usually those are small scale games

which are played in a couple of session and the game state is saved locally on the client. Examples are old LAN style games like for example Quake, Age of Empires, or Diablo 2. If any online matchmaking exists for these games, there is no centralized rule enforcement system in place.

1.5.2 Rich Online Game

These online games only require the player to download the game client or eaven to just play it in the browers. That Game state of all players is stored on the server infrastructure. A part from the game experience comes from this aspect of an online managed game state. The rules must be enforced on the server environment.

1.6 Matchmaking

Matchmaking is a process that helps players to find each other for an online game.

1.7 Problem Research

It would be best if every research would be done completely platform independent. This would inquire very general approaches to document and state the findings. TODO: cite to Lewis Lewis said that if you have implemented something then you have understood it. Since modern programming languages are turing complete it is proven that it is possible to apply the developed concepts also to other programming languages. Question for Advisor: Ist dies legitim? Es wrde mir enorm helfen.

1.8 Delimitations, Limitations, and Assumptions

2 Methods

The research problems that are topic of this thesis are very variegated. For the examination of the problems it is therefor necessary to access each problem in a specifically suited method.

2.0.1 Structure of the Thesis

This thesis is the third part of a three semester project.

- In the first semester (Project thesis 1) is an incorporation into Microservices and the online game domain. It consists of mostly literature research which results in a comprehensive documentation of these topics
- The second semester is about the implementation of game features with Microservices and is devoted to the feasibility study of a Microservice architecture for online game development. A working prototype is the result of the second part.
- The third part is about the elaboration of the previous result
  - A one click solution for developers to start with online game development. The development process should feel as closely as possible to offline game development.
  - Introduction of PCG features to further ease the development of online games.
  - Examination of advanced Microservice topics in this regard such as the deployment of a Microservice cluster with a cluster manager and to evaluate composition of a Microserice environment concretely choreography vs. orchestration. Composition is the glue that keeps a Microservice environment together. A major result of this part of the project is to obtain validation feedback for the solution.

2.1 Subject Pool

2.1.1 Microservice Tenets in Relation to Online Games

2.1.2 Microservice Specific Challenges

2.1.3 Conceptual Problems in Microservice Online Game Envirunments

2.1.4 Microservice Contribution to Online Games

2.1.5 Development of a Microservice Driven Online Game

2.1.6 Procedural Content Generation in Online Games

2.2 Instrumentation

2.2.1 Available Technology

Cloud Service Provider

MicroNet is designed to run in the cloud. It is therefore mandatory to evaluate the major cloud service infrastructure providers according to their usability for online games.

One problem in regards to these service providers is payment. Although they provide a free tier to test their infrastructure, a credit card has to be pledged as security. In the case the limit is overdrawn a fee will be charged. This risk cannot be taken in this thesis.

Google Cloud Gaming Solutions Google offers a solution to deploy rich online games in the cloud. It basically a setup for the existing google cloud infrastructure with additional functionality to to provide build in matchmaking matchmaking. For this purpose it provides an API that is available in a number of programming languages.

The MicroNet service that provides the matchmaking functionality has to be altered if the Google solution for matchmaking wants to be used.

To operate a game driven by MicroNet in the Google Cloud the Google App Engine can be used to run the services of the MicroNet framework and the Google Compute Engine can be used to run the game simulation servers. ActiveMQ that is used for messaging can be used with Google Cloud by using Bitnami.

Amazon Game Lift Amazon GameLift is very similar to Google Gaming Solutions. It also provides a matchmaking solution and the option to communicate with the game backend running in the Amazon cloud. An advantage of the amazon solution is that it provides an API for all the major game engines like Unity3D or Unreal Engine 4. Amazon also provides its own game engine with the name Lumberyard.

A Dedicated Server The most general way to deploy the MicroNet framework is on a dedicated server hardware. This could be either a cluster of bare metal servers with

a sufficient networking capabilities or a set of virtual servers running somewhere in the cloud. Either way the result is a set of hosts running linux and have a public ip address.

For this thesis a virtual server at the HSR will be the test setup. This infrastructure is also available at the major cloud service providers.

Docker Docker is a very powerful paradigm to run software. Instead of installing a software on a host and running it native it is packaged in a container and a container engine on the host is running it instead. This greatly eases the deployment of software since the developer can run it anywhere as long as it runs in Docker.

Docker is the foundation for the easy deployment of MicroNet. It is used to hide all the dependencies that the framework has from the developer. This is possible because to all major dependencies of the Framework dockerhub provides reference containers that can be used to deploy features. This goes for all dependencies namely speaking: ActiveMQ containers and Postgres containers. Since the framework is written in Java developers can simply use openJDK containers to easily deploy their own services.

It is also possible to let a Microservice participate in the framework that is not running in docker. in that case it can simply communicate with the framework over the message broker from any host.

Another interesting docker container is hydra which provides a OAuth2 and OpenID implementation. It can be used to make the framework and the game more secure. But this might be overkill for the system that is aimed to be small and simple. It is easier to use the ActiveMQ security functionality instead.

Docker Compose Docker Compose is a way to describe a complex application consisting of multiple Docker containers. This approach is used to deploy the whole MicroNet framework with one click. Is is possible to automatically update the docker-compose file at compile-time and therefore keep the deployment description in sync with the framework code.

Docker Compose is also the foundation for the usage of Docker Swarm. With docker Swarm a Docker application can be distributed on multiple hosts.

Docker for Windows One requirement for game development is the support for Windows development machines. Since Windows is the most-widely used development OS for games it is mandatory that MicroNet can be fully developed with a Windows machine.

The Docker for Windows Version feels very close to the native version on Linux. One problem regarding Docker for Windows is that it does not run native an on all Windows versions since HyperV is required for it to work. As a work-around Tocker Tools can be used but this makes the deployment process more complicated.

Maven

Apache Maven is a build tool for Java Programs. It defines a build process that includes running tests and finally deploying to program in a repository. Maven helps to build a bridge between the actual code of a service and the integration into the framework. For this purpose it a build script format that defines the dependencies and additional build instructions. This makes it possible do abstract the manual installation process of MicroNet.

So spare the developer of an online game to cope with the edit complicated Maven scripts (.pom files) these pon files are generated by the eclipse MicroNet tools.

2.2.2 Design Science Methodology

2.2.3 Vertical Slice Prototype

2.3 Procedures

2.3.1 Lab Research

2.3.2 Eclipse

Since MicroNet makes usage of various thirs party libraries and tools there needs to be a way to manage them all. This purpose solves the Eclipse tools for MicroNet. The MicroNet Tools is a set of plug-ins that are partially independent of each other. They all help the developer to implement game specific Microservices.

Since the UI of an Eclipse plug-in is written with the SWT library it can easily be exported to a standalone tool that does not require Eclipse. However any code related feature do require the Eclipse IDE.

Service Generation

Allows the developer to not worry about boiler plate code of setting up the service in the framework. This covers the setup of the appropriate networking solution according to the environment. This allows the developer to focus on the actual service code.

Code Assist

The Code Assist plug-in helps the developer to keep track of functionality provided by other services. It allows a type-safe communication between services. The information that is needed to give these API proposals is extracted from the service implementation at compile time. The information is then shared via the version control system. This process is manually and the developer must check in and probably merge his changes.

The format of the distribute API description is not relevant because it is a machine-readable format and the developer sees it in a polished form.

Management UI

The management UI plug-in helps the developer to get an overview over the whole game application. It helps the manage the versions of services of the game services as well as the versions of dependencies. This plug-in highly relies on the description of the maven projects.

2.3.3 Field Research

2.4 Statistical Analysis

The results of the previous semesters are to this point only of hypothetical nature. The main goal this semester is therefore to validate the findings up to now.

Validation for the run-time aspect of the solution:

1. To prove that the solution holds in regard to run-time is necessary to give proper validation feedback for the combination of Microservices and Online Games. It must be shown the running framework is able to scale at run-time. In a first instance this is done by performing lab-research in the form of small scale play test with some elected test players.

2. One more sophisticated approach is to perform TAR on the run-time validation. This would mean to test the MicroNet with the alpha version of the Spacegame prototype in a long running tests that is available for the public.

Validation of the development-time aspect of the solution:

1. It must be proven that the proposed development solution is actually usable and as simple as proposed.

This validation research is made according to design science methods and teminology.

3 Results

3.1 Solutions to the Research Problems

3.1.1 Order of Presentation

3.2 Descriptive Analysis

3.2.1 Microservice Tenet Specific Solutions

3.2.2 Contradictions of the Solutions in a Pure Microservice World

3.2.3 Description of the MicroNet Framework

Microservices reduce the complexity of an online game by splitting the game domain into small shards that can be developed independently. These shards are established by a domain driven design (DDD) approach. Each type of Microservice is repsonsible to handle all aspects of the domain shard completly (cohesion). This reduces the need to invest effort in inter service.

How does MicroNet help the developer to develop large online games:

- MicroNet does not reinvent the wheel for online game development
- MicroNet works in conjunction with existing technologies, specifically popular game engines
- MicroNet tries to make the developer aware of the problems that are encountered during the development process
- While using MicroNet the developer learns the process of online game development
- It helps the developers to get the difficult tasks like consistency of a asynchronous system right

What does MicroNet provide in this regard:

- Provides a clear architecture for the game back-end
- Allows to develop individual game features independently
- Provides a simple protocol definition that helps the developers to loosely couple game features
- Offers standard solutions for technical requirements: Messaging functionality and Persistence functionality

4 Conclusion

4.1 Summary of Findings

4.2 Conclusions Drawn my Results

4.3 Recommendations for Further Research

Bibliography