Empires of Avatharia

Group 22

Christopher Raschke
Johan Wessman
Jonas Sand
Jashar Ghavampour
Johan Granberg

1. Who are the users and what problem does the system solve for them?

Empires of Avatharia is an online strategy game that can be easily accessed from any web browser anywhere in the world. It addresses the problem of curing common boredom. The users could be of any gender and any age, but it will most probably attract strategy interested males between the ages of 15 to 30. We decided to aim specifically for that group since it is common knowledge that this is the group with the most "gamers" in it. Even though this game is going to be easy to learn we hope that it could produce scenarios that even more "serious gamers" will find challenging.

2. The main uses of the system.

The main use of our system is a game meant to be played for fun. It is a strategic game played in any standard web browser making it very flexible to use at most computers with an internet connection. Here are some user scenarios that are meant to illustrate how different users might experience "Empires of Avatharia".

Case 1:

Stig, a twenty-five year old computer interested man, is currently visiting his grandmother over the weekend. He has not brought any computer with him and has therefore no access to any of the games he usually plays. His grandmother's computer is old and slow and can't handle any new games. Fortunately, Stig knows of the game called "Empires of Avatharia", where he is a member. He decides to log in and play some before the family dinner. When logging in he resumes control over his village and army, making new decisions in the game. When he is done he can log out which saves his progress to the server. Stig likes to play "Empires of Avatharia" because it is easily accessible and because it is possible to play as much or as little as one may want.

Case 2:

Alexander, a seventeen year old boy, is currently on a break between classes at school. He has nothing to do and decides to play some "Empires of Avatharia", which he is playing on his free time. The game is easily accessible from the schools computers. He logs in on the website and resumes control over his village, reading a battle report and sending his army on a new attack. After a while his break is over and he logs out and makes his way back to the classroom. He likes to play "Empires of Avatharia" because of the easy access, and being able to play it on short breaks at school. A lot of his friends also play the game which makes it that more fun for him.

Case 3:

Anders is at work and working on his computer. After working a while he decides to take a short break. On his work computer he can easily log in and play "Empires of Avatharia". He likes to play "Empires of Avatharia" because of the easy access and he thinks it is a perfect way of taking a break from work. When he is logged in he manages his village, reads some old battle reports and moves his army. When he is done he logs out, and saves the progress to the server which then takes over control, and he returns to work.

3. **The context/environment in which the system is to be used.**

The game isn't designed for any specific environment as we made clear in the section above. It could be used in a home environment at any time, a school environment at breaks between classes or even at a work-related environment when the employees are bored or find themselves out of work. It's a very flexible game and therefore the environments in which it is used tend to reflect that flexibility.

The game doesn't require any expensive hardware or any complicated installation. It should be easy to access and shouldn't require anything else than a platform, connected to the internet, that is able to interpret and display html code as well as JavaScript. This includes many modern mobile phones and most PC:s etc. The fact that the servers will be up and running "twenty-four seven" also adds to the flexibility of the environment in which the game could be used.

4. The scope of the system.

This is our In/Out list where we point out what we want the project to involve. The list may be changed during the course of the project if new questions arise as the work goes along.

Topic	In	Out
HTML based graphics	X	
Realtime strategy	X	
Login and account management	X	
Communications system between users	X	
Platform independence	X	
Animated graphics		X
3D Graphics		X
2D Graphics	X	
Requires constant presence by user		X
Only the user himself has access to his virtual assets	X	
Computer controlled players		X
Possibility to add AI in the future	X	
Music		X
Game sound		X
Limited number of players	X	
Multiple languages		X
Supervision by administrators		X
Database driven transactions	X	
All calculations are made on the server	X	

Javascript for dynamic handling	X	
Ranking system	X	

5. The main factors that need to be taken in to account when designing and building the system.

One thing that may give us a lot of problems is “unreliable users”. Many of the users in our game are likely to be members of multiple online communities and will therefore have many account names and passwords to remember. The more passwords and account names you use the more likely are you to forget one. Therefore we need a system that can handle that problem without compromising game-account security. This could be done in a few different ways but one solution is to bind an account to an email address.
Another problem that may arise when dealing with unreliable users is the scenario where users are trying to exploit the game. Either they try outright cheating/abusing bugs or just using actions that makes the game imbalanced. “Ingame”-imbalance could be easy handled through patching and monitoring the “ingame-enviroment”. Cheating/abusing bugs on the other hand is harder to handle. We need too make sure that all the user can do is ask the server questions and ask it to do things for them. If we let the users do anything that might affect others on their computers we’re in for a lot of trouble. There are also the odd case of online-vandals that are just out to shut down our game and cause us as much trouble as possible. That need to be addressed with smart database handling methods, to make sure no information is lost even if we have a server crash or something like that. Two ways of fixing this problem is using logs for everything that happens to the server, that way we could just restart the server and update everything that needs updating according to our logs. Another way to do it could be to run parallel database servers.

Another thing to keep in mind is that everybody is not going to start playing our game at the same time, and thus we need to allow people to be able to have a decent chance at growing even if they didn’t join in when the server started.

6. Technologies and Risks

With the game being web based, our choices are narrowed down to a handful of technologies, given that we want to stay with the ones that have proven their worth on a large scale over the past few years. The most obvious available choices under these criteria are PHP, Java or .NET, with MySQL or PostgreSQL databases. They are all nowadays cross-platform.

6. 1 Software

Most members of our group have enough knowledge of and experience with Java to work on the backend, and since Java is an established, reasonably stable, platform independent and widespread language, it is our main language of choice for this project. The mentioned attributes facilitate teamwork and enable us to distribute tasks efficiently, while reducing the risk of a total halt due to any long-term absence of a single member.

The Java Servlet API will be used for this, and that means that the front-end language will be JavaServer Pages, JSP. The familiarity in the group with Servlets and JSP is not extensive, but the time to acquire enough knowledge to be able to do something useful is a matter of hours if the Java skills are good enough.

Most of the HTML and Javascript skills needed for the web interface are concentrated to one or two members of the group, but the simple nature of these languages, along with the limited complexity of the interface, allow for all members to work with the interface using the many web resources available, albeit more time-consuming than for the more experienced.

We will use the PostgreSQL database as most of us have experience with it.

6.2 Hardware

The application itself doesn’t pose any demanding requirements on the hardware on neither the client nor the server side, but with a growing number of users there will be a heavier load in terms of disc access and memory utilization on the server side, which will require upgrades of existing servers, expansion to new servers, load balancing and so on. A modern home computer will accommodate at least a few hundred simultaneous users with ease so we will code with scaling in mind, but hardware upgrades are probably no concern until the number of users grows far beyond our group. The natural stops in the game allow for certain downtime, allowing maintenance and upgrades without a heavy impact on the user experience.

We will probably not have different servers for development, testing or staging while the game is under development, this is for practical and financial reasons. We will try to use some revision control system such as Subversion to keep the code gathered, accessible, updated and structured. This is all likely to be done on the same computer, perhaps with a web hosting company or somehow through KTH.

6.3 Risks

Even with its limited complexity, the greatest concern is still the web interface coding and the Servlet behind it. All members will have to acquire some new skills to be able to work efficiently as a team, and this may take too much time if it is not well coordinated from the start.

Using the standard, well tested frameworks, we have much less to worry about when it comes to stability, but there are still concerns and it’s important to patch everything as soon as possible to avoid security issues with newly discovered vulnerabilities. This may be difficult when not having complete control of the hardware, as is the situation with web hosts, a very important fact to keep in mind if any expansion of the game is needed.

The infrastructure around the server is also important to keep in mind if things get more serious – a real fireproof server hall with redundant electricity and physically separated connections to the net is much better than having a server standing in the basement of some office.

Backups are crucial for us right from the start and we have to make sure we have updated copies of everything on the server in case something unexpected would happen, especially in the development phase.