Systemy kontroli wersji:

W wyborze systemu kontroli wersji głównym czynnikiem w podjęciu decyzji z którego będziemy korzystać była aktualna biegłość w obsłudze jakichkolwiek systemów przez członków mojego zespołu. Po sprawdzeniu aktualnie polecanych systemów kontroli wersji okazało sie, że każdy z członków zespołu potrafi posługiwać się występującym w większości artykułów repozytorium **Github.**
https://github.com/Palodiuss/IO

Drugim repozytorium które używaliśmy było repozytorium http://redmine.cel.agh.edu.pl/ Na tym repozytorium każdy z członków zespołu miał dostęp do listy zadań które należało wykonać z odpowiednim priorytetem oraz podgląd postępów.

Język programowania:

Początkowo nasz wybór padł na język **C++** z oczywistego względu – poziom znajomości. Każdy z członków naszego zespółu najlepiej czuje się w tym właśnie języku. Jego funkcjonalność w zupełności wystarczała do naszego zadania. Niestety nasz wybór musiał zostać zmieniony poprzez kilka czynników:

- Pozostałe grupy na swój język programowania wybrały język JAVA co utrudniało finalne połączenie modułów programu w sytuacji gdy jesteśmy jedyną grupą która nie pisze w tym języku
- Ostateczna wersja programu miała łączyć się z bazą danych co jest łatwiejsze do zaimplementowania w języku JAVA
- Jeden z naszych programistów jest biegły w tym języku przez co podjeliśmy decyzję o zmianie naszego wyboru na język JAVA

IDE:

Jako IDE wybraliśmy **InteliJ Idea**. Jest to najbardziej zaawansowane IDE dla Javy na dzień dzisiejszy. Działa na wszystkich systemach operacyjnych oraz posiada dużą bazę wysokiej jakości wtyczek. Minusem jest fakt iż jest to oprogramowanie płatne lecz w naszym
przypadku posiadamy uczelnianą licencję co pozwala nam na użytkowanie pełnej wersji programu bez opłat.

Wymiana danych:

Do wymiany danych wybraliśmy format **JSON**. Jest to prosty format tekstowy do przesyłania danych, który posiada parsery we wszystkich najpopularniejszych językach. Format ten jest bardziej przejrzyty od jego największego konkurenta XML a dodatkowo zarówno nasi programiści jak i innych grup są z nim dość dobrze zaznajomieni.

Modelowanie UML:

W przypadku wybrania narzędzi do modelowania UML posłużyliśmy się dwoma rozwiązaniami. Pierwsze z nich to płatne rozwiązanie **Visual Paradigm.** Program ten jest podany na stronie prowadzącego oraz był wykorzystywany na zajęciach więc członkowie zespołu mieli okazję się z nim zapoznać. Podczas instalacji wystarczyło podać email na domenie @student.agh.edu.pl aby otrzymać kod do aktywacji pełnej wersji programu.
Drugim z rozwiązań była darmowa aplikacja online **draw.io**. Jest ona bardzo prosta w obsłudze a dodatkowo niektórzy członkowie zespołu są bardzo biegli w jej obsłudze co ułatwiło ich pracę.

Komunikacja:

Podstawowa komunikacja zarówno wewnątrz grupy jak i pomiędzy innymi grupami opierała się na wysyłaniu wiadomości na portalu społecznościowym. Została utworzona konferencja grupowa ze wszystkimi członkami grupy gdzie wymienialiśmy postępy w pracy oraz spostrzeżenia. Gdy potrzebowaliśmy omówić coś dokładniej komunikowaliśmy się przy użyciu aplikacji **Skype**.
Komunikacja pomiędzy grupami oraz wymiana danych nie była już taka łatwa, zdarzało się iż inne grupy nagle zmieniały wcześniej ustalony format danych lub po prostu zalegały z dostarczeniem nam odpowiedniej ilości informacji co bardzo opóźniało pracę naszych programistów.

Bibliografia:

Systemy kontroli wersji
https://biz30.timedoctor.com/git-mecurial-and-cvs-comparison-of-svn-software/
IDE:
http://www.javaworld.com/article/3114167/development-tools/choosing-your-java-ide.html
UML:
https://www.visual-paradigm.com/
http://www.pcmag.com/review/350373/draw-io