Projekt TransFiler

January 5, 2016

Paweł Lewko
Tomasz Bartkowski
Tomasz Bigorowski
Michał Kordel
Jan Tatarynowicz

Projekt TransFiler

1 Wstęp

W ramach projektu zostanie utworzony system o nazwie kodowej `TransFiler`. System `TransFiler` będzie aplikacją mobilną na urządzenia z systemem mobilnym `Android`, służącą do przesyłania plików za pomocą dźwięku.

2 Zakres

Praca swoim zakresem obejmuje aplikacje mobilne służące komunikacji bezprzewodowej. W przeciwieństwie do typowych rozwiązań tego typu, używających jako medium transmisyjnego fali elektromagnetycznej, projekt `TransFiler` zakłada komunikację przy użyciu fali mechanicznej, a konkretnie dźwięku w zakresie słyszalnym. Aplikacja pozwoli ominąć problemy niezgodności sprzętowych i nie zgodności intefejsów, które często występują przy komunikacji radiowej. Zostanie dobrany odpowiedni zakres częstotliwości przesyłanego dźwięku, aby zapewnić możliwie jak największą prędkość wysyłania przy jak największej niezawodności. Poprawność każdej transmisji będzie weryfikowana kodem `CRC`. Zostanie także dobrana odpowiednia metoda modulacji dźwięku. Aby zapewnić, że dane zostały przesłane bezbłędnie zostaną użyte odpowiednie algorytmy sum kontrolnych. System będzie posiadał Graficzny Interfejs Użytkownika na platformie `Android`. Aplikacja zostanie napisana w języku `Java` przy wykorzystaniu biblioteki `Beads`.

3 Cel

Celem pracy jest zaprojektowanie i oprogramowanie aplikacji mobilnej o następujących założeniach funkcjonalnych:

- wybór dowolnego pliku do wysłania,
- odpowiednie przetworzenie bitów i modulacja fali dźwiękowej tymi bitami,
- odtworzenie fali dźwiękowej,
- nagranie fali dźwiękowej,
- demodulacja fali dźwiękowej i przetworzenie bitów,
- weryfikacja poprawności przesyłanych danych,
- zapisanie pliku w pamięci.

4 Przegląd podobnych rozwiązań

Obecnie na rynku istnieje wiele aplikacji służących bezprzewodowej komunikacji pomiędzy urządzeniami mobilnymi. Rozpatrzmy najpierw te których medium komunikacyjnym jest fala elektromagnetyczna:

`Bluetooth File Transfer` – aplikacja na system `Android` firmy `Medieval Software` używa technologii `Bluetooth`. Istotną zaletą tej aplikacji jest zgodność ze starszymi standardami `Bluetooth`, używanych w czasie gdy wiodącym mobilnym systemem operacyjnym był Symbian OS, dzięki temu możliwa jest komunikacja również z tymi urządzeniami. Przebieg wysyłania pliku wygląda następująco: wybieramy plik do wysłania, następnie skanujemy w poszukiwaniu urządzeń z aktywnym interfejsem `Bluetooth`, a gdy wybierzemy urządzenie do którego chcemy wysłać plik oczekujemy na akceptację wysyłki. Aplikacja posiada wbudowane szyfrowanie `AES` (128, 192 i 256-bitowe). Poprawność przesyłanych może zostać zweryfikowana przy użyciu sum kontrolnych `MD5` i `CRC32`.

`Wifi File Transfer` – stworzona przez firmę `smarterDroid` aplikacja mobilna umożliwiająca przesyłanie plików korzystając z protokołu `WiFi`. Zwykle nie wysyła ona danych bezpośrednio z jednego urządzenia na drugie, lecz przez punkt dostępowy sieci – zalogowanie do tej samej sieci `WiFi` jest wymagane. Możliwe jest jednak uruchomienie w urządzeniu trybu `Hotspot`, wówczas staje się ono punktem dostępowym i pliki mogą być przesyłane bezpośrednio. Na urządzeniu z którego mają zostać wysłane uruchomiony zostaje serwer `HTTP`, widoczny tylko w obrębie lokalnej sieci. Ma to istotną zaletę, mianowicie odbierający pliki nie musi instalować aplikacji, wystarczy jedynie przeglądarka internetowa. Ma to jednak również wadę ponieważ serwer pracuje na niestandardowym dla protokołu `HTTP` porcie 1234, a jak wiadomo w niektórych sieciach `WiFi` niestandardowe porty są zablokowane.

`ShareCloud` – rozwiązanie firmy `For2ww`, aplikacja umożliwiająca nie tylko przesyłanie plików ale również przechowywanie ich w chmurze. Od użytkownika wysyłającego jak i odbierającego wymaga dostępu do Internetu, co może być niekiedy utrudnieniem, jednak ma to tę zaletę, że pliki mogą być wysyłane bez względu na dystans pomiędzy urządzeniami. Dodatkowo odbieranie nie musi zachodzić w chwili wysyłania, ponieważ dane w mogą być przechowywane w chmurzę przez dowolny czas.

Jeśli chodzi o przesyłanie plików przy użyciu fali mechanicznej, to istnieje aplikacja `Chirp` (stworzona przez firmę `Animal Systems`) która jest najbliższa koncepcji przesyłania plików przy pomocy dźwięku i jest promowana jako taka właśnie aplikacja. Nie jest to jednak do końca prawdą, ponieważ wymaga ona do działania dostępu do Internetu, ponieważ w rzeczywistości dane przesyłane są przez Internet (plik jest wysyłany do serwera, a następnie pobierany z serwera przez urządzenie odbierające). Dźwiękiem przesyłany jest jedynie identyfikator pliku potrzebny do pobrania pliku z serwera.

Aplikacja `TransFiler` będzie zatem jedyną aplikacją mobilną umożliwiającą przesyłanie danych jedynie przy użyciu fali mechanicznej.

5 Analiza problemu

Aplikacja `TransFiler` służy do przesyłania plików za pomocą dźwięku. Używa zatem fali mechanicznej, a nie jak większość tego typu aplikacji - fali elektromagnetycznej. Istnieje aplikacja mobilna `Chirp`, która co prawda używa dźwięku do przesłania plików, jednak nie może działać bez dostępu do Internetu - dźwiękiem przesyłane są jedynie dane potrzebne do pobrania pliku z serwera. `TransFiler` nie potrzebuje dostępu do Internetu, do przesłania pliku wystarczy, że urządzenia będą posiadały głośnik i mikrofon.

Aplikację po uruchomieniu będzie można przestawić w jeden z dwóch trybów - tryb nadawania (`Player`) i tryb nasłuchiwania (`Recorder`). W trybie nadawania możemy wybrać plik z pamięci urządzenia i rozpocząć wysyłanie. Aby plik mógł zostać przesłany najpierw musi zostać prrzetworzony na ciąg bitów (komponent `File Converter`), a następnie zmodulowany. Do modulacji posłuży biblioteka dźwiękowa `Beads` napisana w języku Java i wydana na licencji `Open Source`. Użytą metodą modulacji będzie `Frequency Shift Keying`. Jest to modulacja częstliwościowa, generuje ona sygnał sinusoidalny o mniejszej częstotliwości dla wartości 0 i sygnał o większej częstotliwości dla wartości 1. Ma ona tą przewagę nad modulacją amplitudową że niewielkie zmiany amplitudy, spowodowane np. zmianą dystansu między urządzeniami nie będą powodowały błędów transmisji. Dodatkowo jest stosunkowo prosta w implementacji.

Planowany kod modulacyjny:

$$\alpha(x_n) = \begin{cases} A_0 sin 500t & dla \quad x_n = 0 \\ A_0 sin 1000t & dla \quad x_n = 1 \end{cases}$$

Wartości częstotliwości mogą się zmienić , jeśli w czasie testów okaże się że częstotliwości 500Hz oraz 1000Hz wcale nie są najlepsze. Z twierdzenia Shannona-Hartleya wynika, że zwiększając częstotliwość będzie można uzyskać większą przepustowość, dlatego zostaną przetestowane w szczególności wyższe częstotliwości. W przypadku gdy zajdzie taka potrzeba, może zostać zaimplementowane dopasowywanie częstotliwości w trakcie działania aplikacji (taka funkcja może być przydatna w sytuacji gdy warunki zewnętrzne do korzystania z danej częstotliwości będą uniemożliwiały poprawną transmisję). Dodatkowo zostaną użyte dodatkowe częstotliwości do ramkowania oraz do zaznaczenia końca transmisji.

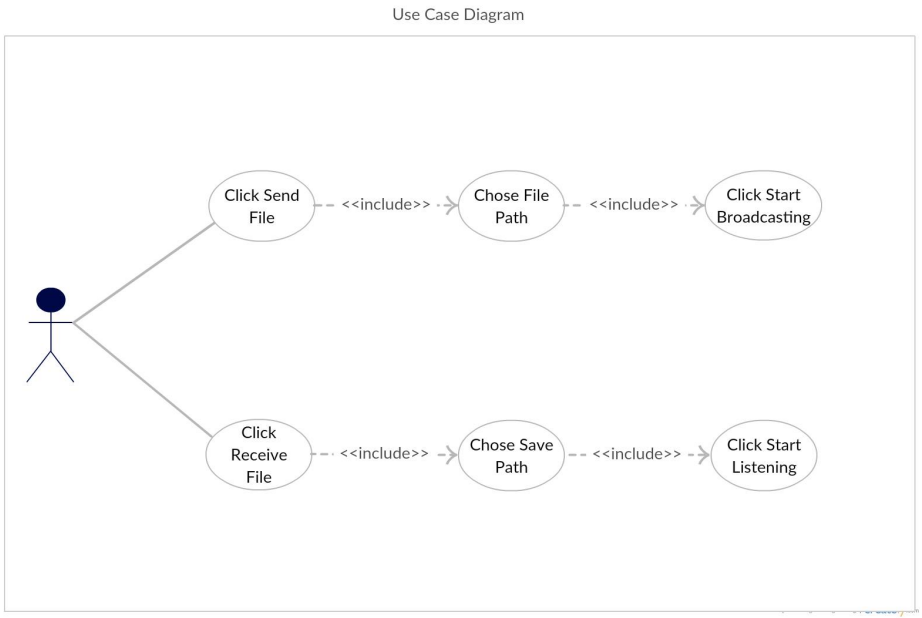
Jeśli cała komunikacja będzie miała poprawny przebieg to fala dźwiękowa na urządzeniu nasłuchującym zostanie poddana demodulacji i po zamianie ciągu bitów na obiekt typu `File` da nam możliwość zapisania pliku w pamięci. Dzięki zastosowani Szybkiej Transformaty Fouriera, która jest zaimplementowana w bibliotece `Beads`, demodulacja fali dźwiękowej będzie wydajna. Jeśli w transmisji nastąpią przekłamania bitów, to urządzenie nasłuchujące wykryje ten fakt i zostanie wyświetlony odpowiedni komunikat. Do sprawdzania poprawności danych zostanie użyty kod `CRC` (`Cyclic Redundancy Check`).

6 Projekt systemu

6.1 Grupy użytkowników i założenia

Program działa na urządzeniach z systemem `Android` z działającym mikrofonem i głośnikami. Użytkownikiem nazwiemy osobę która posiada program z zainstalowanym programem na urządzeniu android. Użytkownicy mogą przesyłać dane między swoimi urzadzeniami. Program nie posiada trybów dostępów, każdy użytkownik ma pełną funkcjonalność. Program posiada interfejs użytkownika(GUI) i działa na zasadzie użytkownik - użytkownik.

6.2 Diagram przypadków użycia



Rys 1: Diagram przypadków użycia

6.2.1 Scenariusze przypadków użycia

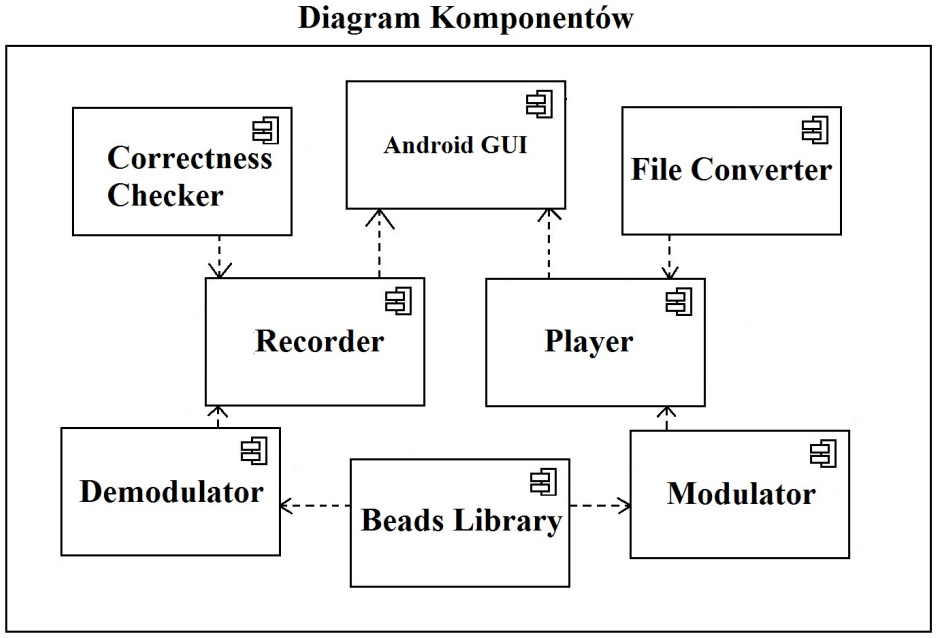
Przepływ zdarzeń między użytkownikiem a programem dla przypadku użycia „Wyślij Plik":

1. Program wyświetla dostępne opcje na ekranie, użytkownik wybiera opcję wyślij plik poprzez kliknięcie palcem w pole/guzik o nazwie wyślij plik.
2. Program wyświetla menażer plików dla użytkownika, użytkownik wybiera ścieżkę do pliku który chce wysłać lub wpisuje ją z klawiatury systemowej.
3. Program akceptuje ścieżkę do pliku następnie wyświetla pole/guzik start, użytkownik klika palcem start aby wysłać plik.

Przepływ zdarzeń między użytkownikiem a programem dla przypadku użycia „Odbierz Plik”:

1. Program wyświetla dostępne opcje na ekranie, użytkownik wybiera opcję Odbierz plik poprzez kliknięcie palcem w pole/guzik o nazwie Odbierz plik.
2. Program wyświetla menażer plików dla użytkownika, użytkownik wybiera ścieżkę do folderu w którym chce zapisać plik lub wpisuje ją z klawiatury systemowej.
3. Program akceptuje ścieżkę do pliku następnie wyświetla pole/guzik start, użytkownik klika palcem start aby odebrać plik.

6.3 Diagram komponentów



Rys 2: Diagram komponentów

`Player` - moduł odpowiadający za odtwarzanie dźwięku zapisanego przez modulator

`Recorder` - moduł odpowiadający za rejestrowanie dźwięku z otoczenia

`Modulator` - moduluje dźwiek w danym zakresie częstotliwości za pomocą `Beads Library`

`Demodulator` - przekształca zarejestrowany dźwięk nagrany za pomocą `Recordera` do postaci binarnej

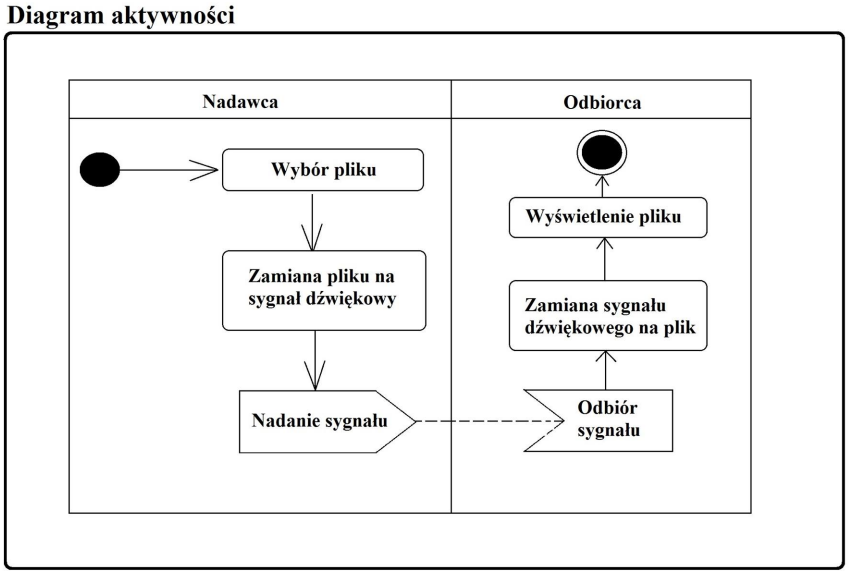
`Correctness Checker` - sprawdza błędy CRC

`File Converter` - konwertuje dowolny plik na postać binarną i na odwrót dodając kod `CRC`

`Beads Library` - biblioteka pozwalająca zapisywać dane binarne w postaci funkcji którą łatwo odtworzyć w `Player`

`Android GUI` - graficzny interfejs udostępniający w intuicyjny sposób wszystkie funkcjonalności systemu użytkownikowi telefonu z systemem

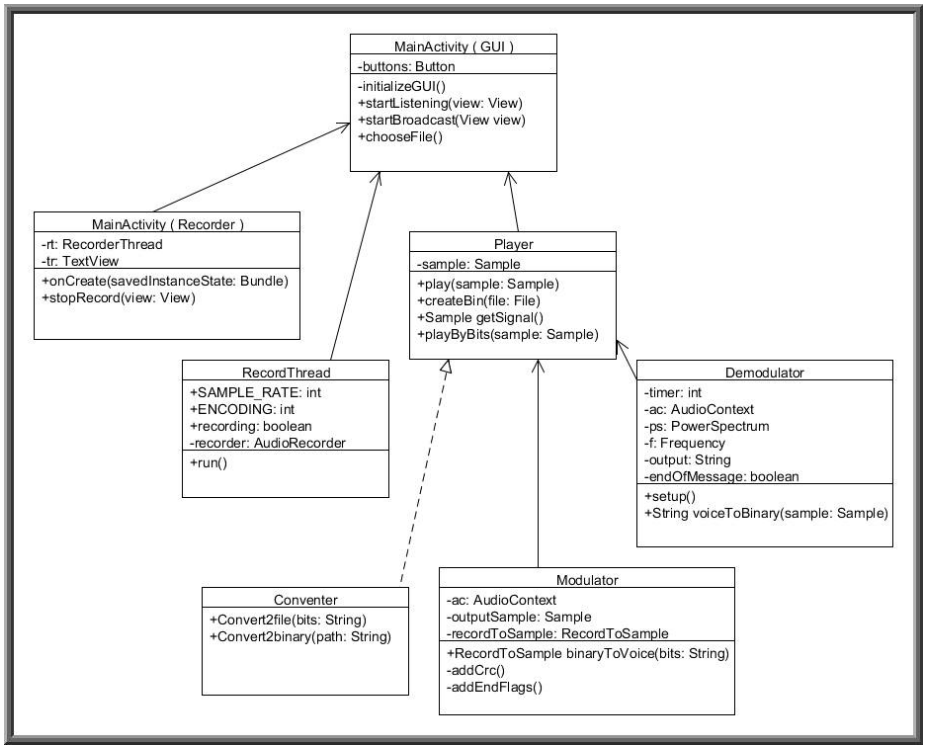
6.4 Diagram aktywności



Rys 3: Diagram aktywności.

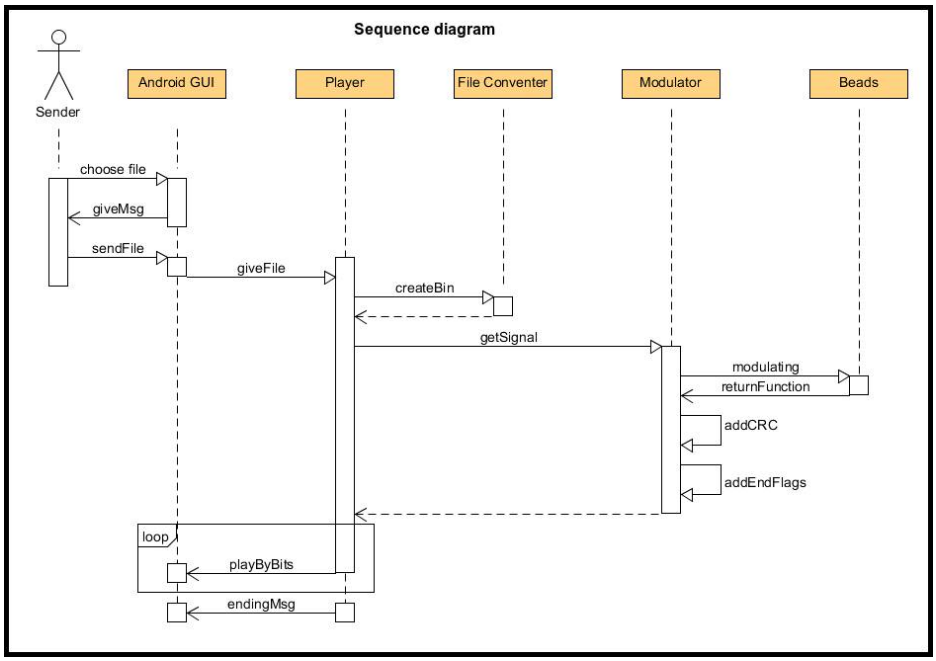
Na początku użytkownik dostaje możliwość wyboru plik. Kiedy plik zostanie wybrany zostaje on przetworzony na ciąg bitów i zamieniony na sygnał dźwiękowy. Po tym następuje nadawanie sygnału. W tym samym czasie drugie urządzenie odbiera sygnał dźwiękowy. System zamienia sygnał na plik, po czym zostaje zapisany on w pamięci.

6.5 Diagram klas



Rys 4: Diagram klas

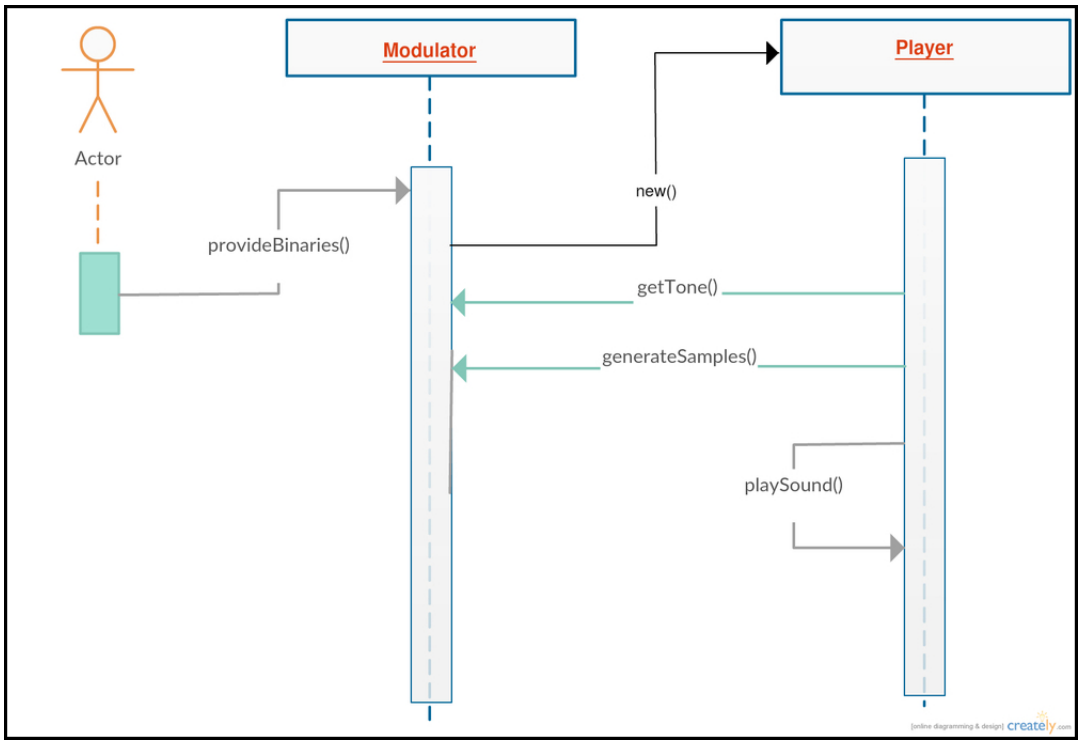
6.6 Diagram sekwencji



Rys 5: Diagram sekwencji dla aktora wysyłającego

Osoba wysyłająca dane wybierając plik wywołuje metodę `chooseFile()`, `Android GUI` zwraca informację o powodzeniu lub niepowodzeniu operacji. Następnie klikając odpowiedni przycisk używa metody `sendFile()` klasy `Android GUI`, która wywołuje metodę `giveFile()` klasy `Player`. Obiekt klasy plik przekazywany jest do klasy `File Converter`, która zwraca ciąg bitów. Następnie `Player` wywołuje metodę `getSignal()` klasy `Modulator`, która odwołując się do biblioteki `Beads` moduluje sygnał, a następnie dodaje kod `CRC` i dodaje sygnał końca transmisji. Sygnał właściwy, sygnał kodu `CRC` i sygnał końca transmisji są połączone i mogą być zwrócone do obiektu `Player`. Następnie `Player` odtwarza sygnał w pętli aż do zakończenia, i zwraca obiektowi `Android GUI` informację o zakończeniu transmisji.

6.7 Diagram sekwencji dla modulatora



Rys. 6: Diagram sekwencji dla aktora modulującego

Do modułu odpowiadającego za modulację dźwięku zostaje przekazany plik binarny. Obiekt klasy `Modulator` tworzy nowy obiekt klasy `Player`, który z obiektu `Modulator` pobiera odpowiednie częstotliwości odtwarzania, a następnie przy użyciu metody `generateSamples()` generuje próbki dźwięku które później zostaną odtworzone. Kiedy mamy wszystkie próbki zostaną wygenerowane dźwięk zostaje odtwarzany metodą `playSound()` klasy `Player` .