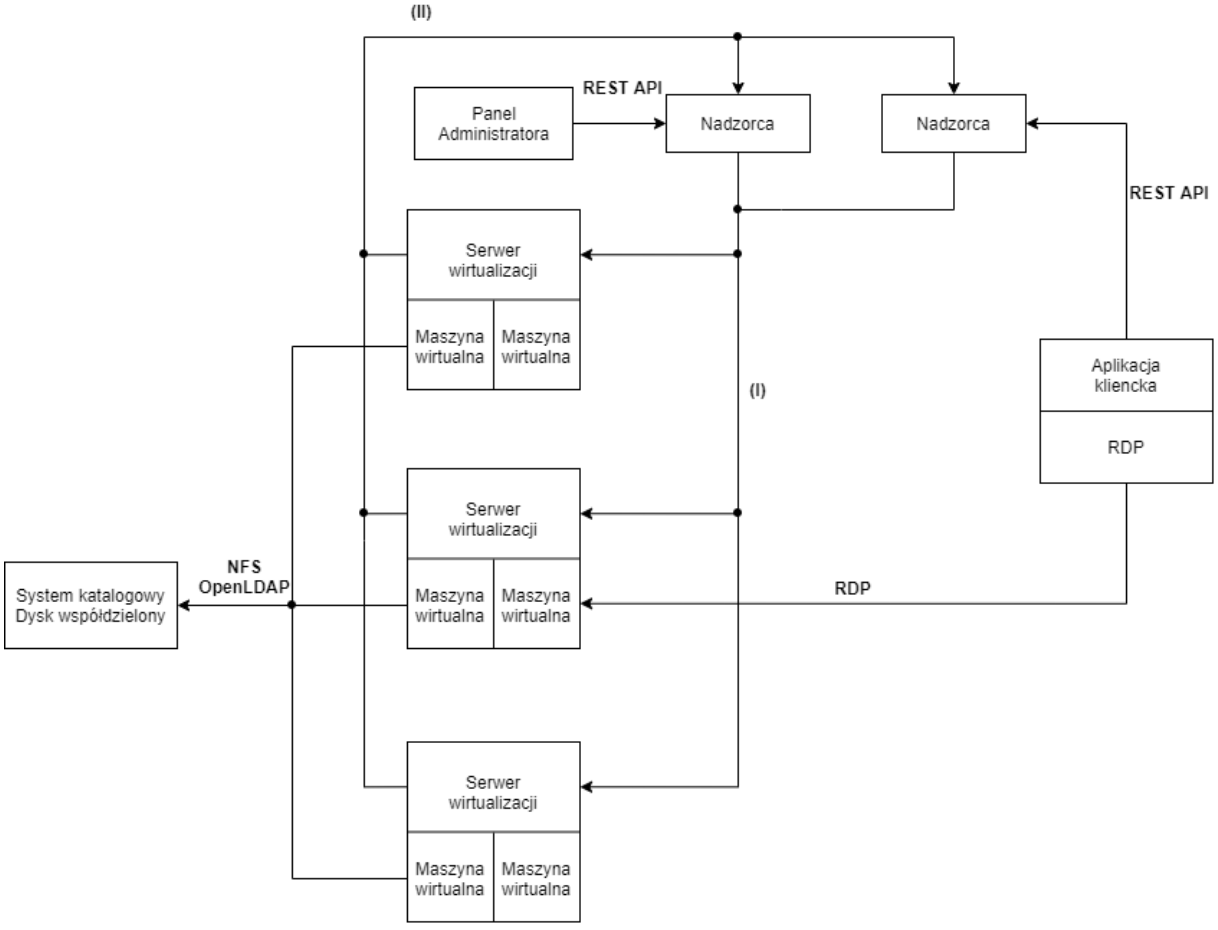
0.1. Architektura systemu

Opracowywany system składa się z następujących modułów:

- nadzorcy,
- serwera wirtualizacji,
- aplikacji klienckiej,
- panelu administratora,
- brokera wiadomości,
- systemu katalogowego,
- dysku współdzielonego.

Schematyczny obraz systemu przedstawia poniższy rysunek.



Rysunek 0.1: Schematyczna architektura systemu

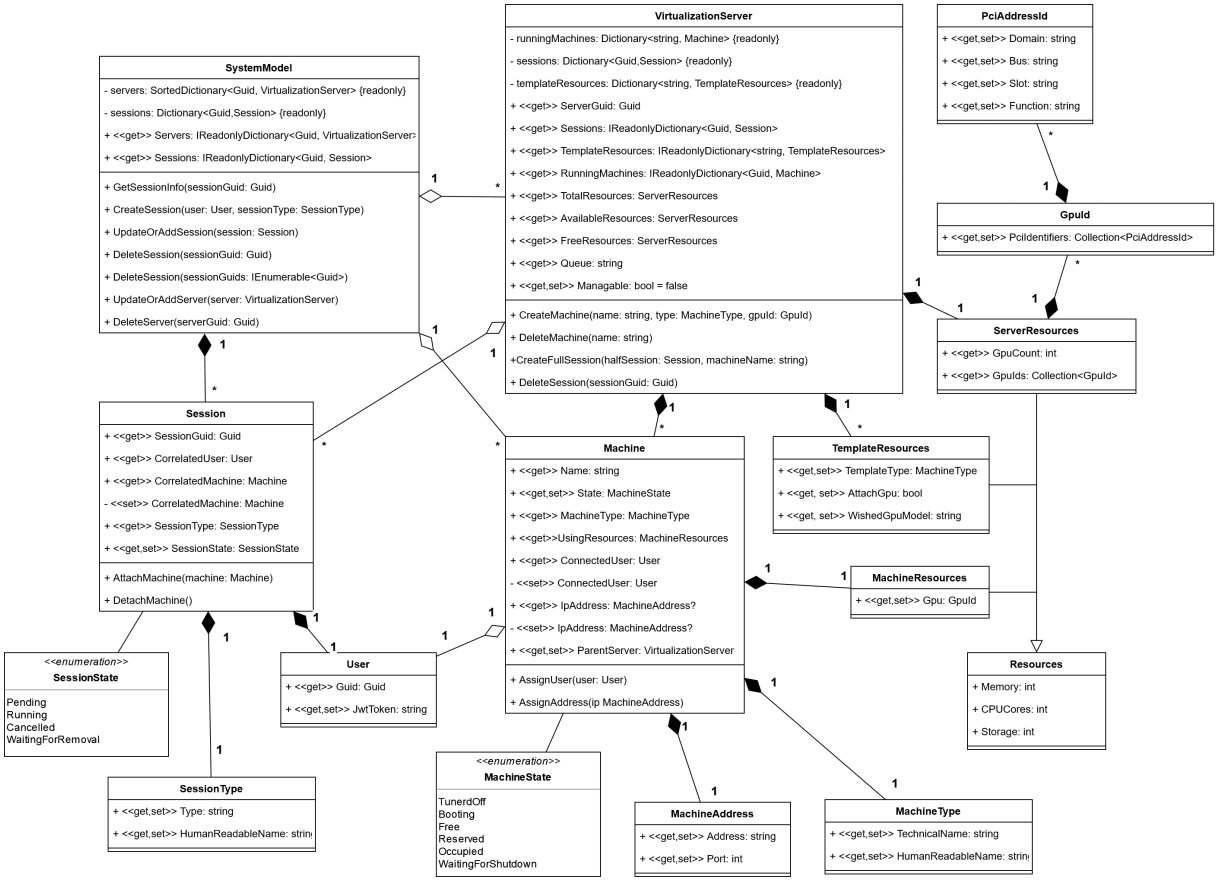
Połączenia oznaczone liczbami rzymskimi oznaczają kolejki komunikacji za pośrednictwem brokera wiadomości, które opisane zostały w punkcie jemu poświęconym. Z założenia system powinien móc skalować się w dwóch wymiarach, to znaczy:

1. Zwiększanie liczby serwerów wirtualnych - zwiększenie liczby istniejących jednocześnie sesji.

2. Zwiększenie liczby nadzorców - zwiększenie liczby obsługiwanych klientów jednocześnie oraz niezawodności systemu.

0.1.1. Model systemu

W celu zarządzania systemem, każdy z nadzorców musi posiadać dokładną wiedzę o jego aktualnym stanie. Informacje te przechowuje w strukturze nazywanej dalej modelem systemu. Ważnym jest, że klasy modelu nie wykonują żadnych akcji poza modyfikacją przechowywanych danych. Wszelkie metody służą jedynie do zmiany stanu przechowywanego modelu w celu dopasowania go do rzeczywistego stanu, na podstawie otrzymanych danych. Przyczyną takiego stanu, jest pierwsze z założeń komunikacji opisane w 0.5.1. Sprawia ono, że zmiana modelu musi być następstwem pewnej akcji wykonanej przez serwer wirtualizacji.



Rysunek 0.2: Schemat klas modelu systemu

Główną klasą modelu jest `SystemModel` zawierający informacje o aktualnie działających serwerach wirtualizacji oraz aktywnych sesjach. Klasa `VirtualizationServer` modeluje pojedynczy serwer wirtualizacji, jego zasoby, maszyny wirtualne oraz obsługiwane sesje. Przechowuje ona również informacje wymagane do komunikacji z daną instancją serwera.

Klasa `Resources` oraz jej pochodne opisują zasoby systemowe i są używane do przedstawienia zarówno zasobów maszyny jak i całego serwera wirtualizacji. Jej klasa pochodna `TemplateResources` służy do przechowywania informacji o zasobach potrzebnych do utworzenia maszyny danego typu.

0.1.2. Nadzorca

Aplikacja mająca za zadanie obsługiwać komunikację z aplikacjami klienckimi oraz wysyłać polecenia do serwerów wirtualizacji. Udostępnia REST API służące do komunikacji z aplikacjami klienckimi. Do komunikacji z serwerami wirtualizacji wykorzystuje kolejki.

Nadzorca przechowuje wewnętrznie model systemu zawierający informację o działających serwerach wirtualizacji i stanie ich maszyn. Na podstawie tego modelu moduł stwierdza, do której maszyny przypisać nowo utworzoną sesję. Wewnętrzne procesy skupione są wokół zmian modelu. Jeżeli proces wysłał do serwera wirtualizacji prośbę o zmianę stanu, to dalsze procesowanie odbywa się, gdy stan modelu został zaktualizowany, i na podstawie jego stanu podejmowane są decyzje.

Dzięki zastosowaniu kolejek oraz zasad komunikacji w systemie może istnieć więcej niż jeden nadzorca. Instancje nadzorców działają niezależnie od siebie i przechowują identyczny model systemu. Dzięki temu uzyskujemy retencję i możemy zmniejszyć obciążenie poszczególnych nadzorców.

0.1.3. Serwer wirtualizacji

Zadaniem serwera wirtualizacji jest uruchamianie i zarządzanie maszynami wirtualnymi, z którymi łączy się użytkownik systemu. Komunikuje się on z nadzorcami i wykonuje operacje na maszynach wirtualnych zgodnie z żądaniami.

Moduł ten nie jest w stanie funkcjonować samodzielnie. Z tego powodu aplikacja nie uruchomi się, jeżeli nie jest w stanie nawiązać połączenia z aplikacją nadzorczą, a w przypadku ostatni nadzorca w systemie zakończy działanie, aplikacja również je zakończy, pod warunkiem że nie ma żadnych działających sesji.

Serwer wirtualizacji jest częścią systemu, która przechowuje realne zasoby udostępniane użytkownikom. System zaprojektowany jest w taki sposób aby teoretycznie nie było ograniczenia na

liczbę serwerów wirtualizacji działających jednocześnie.

0.1.4. Aplikacja kliencka

Aplikacja okienkowa umożliwiająca użytkownikowi autoryzację, uzyskanie sesji oraz automatyczne rozpoczęcie połączenia. Komunikuje się z nadzorcą za pomocą REST API.

Proces uzyskania sesji z perspektywy aplikacji klienckiej zawiera:

1. Uzyskanie informacji o dostępnych typach i liczbie maszyn
2. Wybór typu maszyny
3. Oczekiwanie na utworzenie sesji
4. Nawiązanie połączenia RDP
5. Utrzymanie i monitorowanie stanu połączenia.

0.1.5. Panel administratora

Prosta aplikacja internetowa umożliwiająca administratorowi systemu podgląd stanu zużycia zasobów serwerów wirtualizacji.

0.1.6. Broker wiadomości

Komunikacje wewnątrz systemu, czyli pomiędzy serwerami wirtualizacji oraz nadzorcami, będzie realizowali poprzez kolejki wiadomości. W tym celu użyty został system RabbitMQ, który zajmuje się transportem wiadomości wewnątrz systemu oraz niezawodnością komunikacji między modułami.

Zdefiniowane zostały następujące kolejki wiadomości:

(I) Kolejka kończąca się na każdym z serwerów wirtualizacji powielająca wiadomości między nich. Służy ona do wysyłania nie spersonalizowanych próśb od nadzorców do serwerów wirtualizacji.

(II) Kolejka kończąca się na każdym z nadzorców powielająca wiadomości między nich. Służy ona do przesyłania informacji do nadzorców o zmianach wewnątrz serwera wirtualizacji.

(III) Kolejka kończąca się wyłącznie na pojedynczym serwerze wirtualizacji. Liczba kolejek zgadza się z liczbą serwerów wirtualizacji aktywnych w systemie. Służą one do przesyłania spersonalizowanych wiadomości oraz sprawdzania, czy serwer wirtualizacji nadal pracuje po drugiej stronie. Skorzystamy z funkcjonalności kolejek na wyłączność(Exclusive Queue[1]).

[1]Opis zachowania kolejek na wyłączność

(IV) Kolejka kończąca się na aktualnie podłączonym do maszyny wirtualnej kliencie. Podobnie jak powyżej kolejek istnieje tyle ile aktywnych użytkowników. Celem kolejki jest sprawdzenie, czy aplikacja kliencka nadal jest podłączona do wirtualnej maszyny (mechanizm Exclusive Queue). W celach bezpieczeństwa będą one definiowane na oddzielnym procesie brokera, który będzie można w razie potrzeby udostępnić poza sieć lokalną.

Powyższe 4 grupy kolejek umożliwią prawidłowe działanie systemu. Każdy z modułów tworzy w trakcie uruchamiania kolejki, z których odbiera wiadomości. Jedynym wymogiem prawidłowego uruchomienia komunikacji jest dostępny dla wszystkich serwerów wirtualizacji oraz nadzorców proces brokera.

0.2. Zewnętrzne narzędzia

0.2.1. Ansible

Ansible został wykorzystany w systemie do zaaplikowania zmiennej konfiguracji do każdej uruchamianej wirtualnej maszyny. Podstawowo playbook będzie zawierać informacje o:

1. danych dostępowych do dysku sieciowego oraz wykorzystanym protokole,

2. danych dostępowych do usługi katalogowej.

Playbook można rozszerzać o potrzebne dane zależne od użycia.

0.2.2. Vagrant

Vagrant został wykorzystany w celu łatwej parametryzacji oraz powtarzalnego tworzenia maszyn wirtualnych z przygotowanego wcześniej obrazu systemu.

Wykorzystywany jest głównie mechanizm Vagrant-boxów, które są obrazami wcześniej przygotowanego systemu operacyjnego. Aby system działał prawidłowo obraz systemu zamknięty w Vagrantboxie musi spełniać następujące warunki:

1. Użytkownicy muszą być pobierani z usługi katalogowej.

2. Katalogi domowe użytkowników muszą być na dysku sieciowym.

3. Musi istnieć serwer RDP

0.2.3. Libvirt z QEMU

Libvirt połączony z QEMU jest wykorzystany do zarządzania maszynami wirtualnymi uruchamianymi na serwerze wirtualizacji. Umożliwia on:

1. Tworzenie maszyn wirtualnych.

2. Uruchamianie maszyn wirtualnych.

3. Przyporządkowanie zasobów maszynom wirtualnym (w tym krat graficznych).

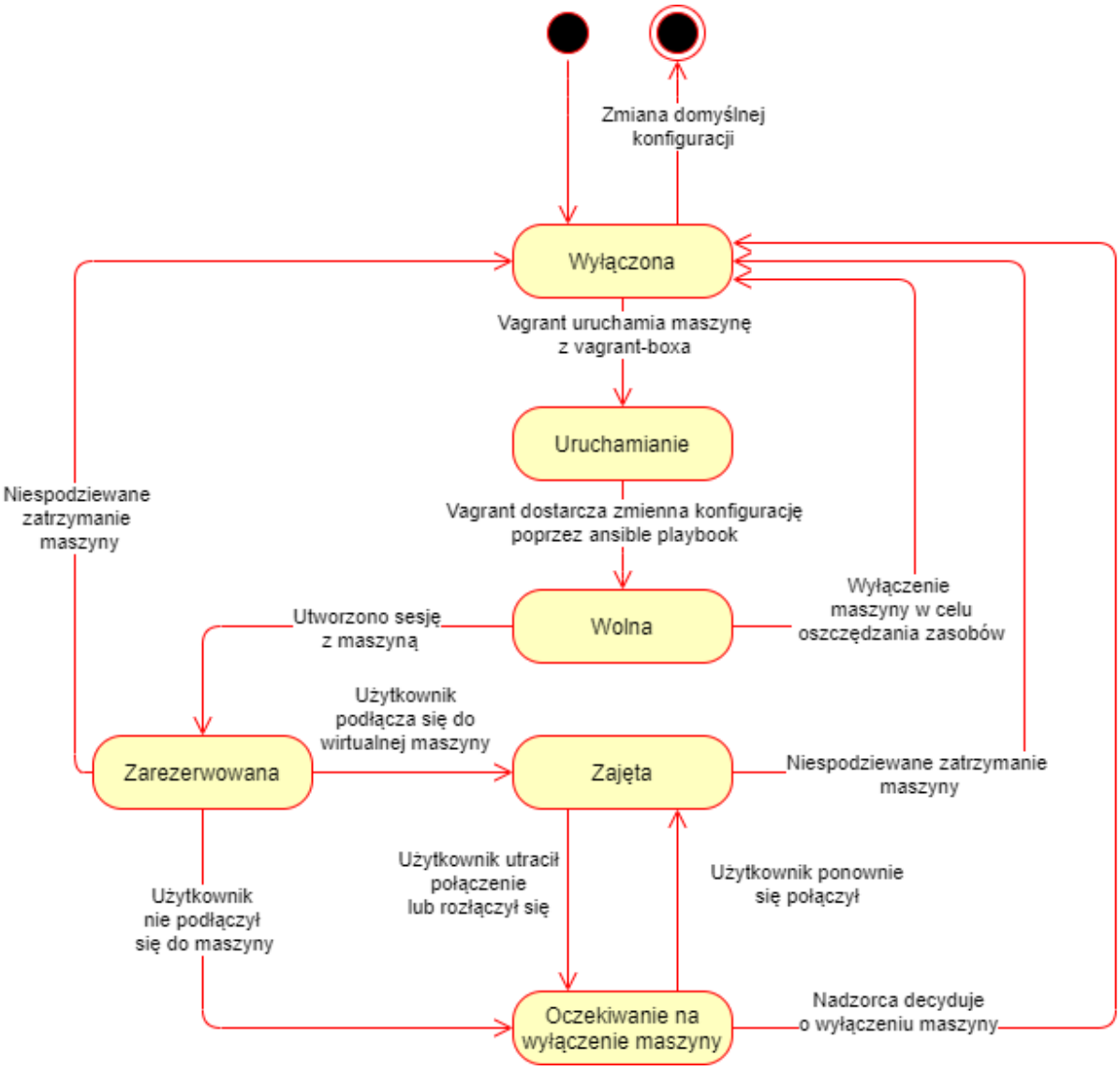
4. Wyłączanie maszyn wirtualnych.

5. Sprawdzanie, czy maszyna o danej nazwie już działa.

0.3. Stany biznesowe

0.3.1. Maszyna wirtualna

Najważniejszym obiektem biznesowym w systemie jest maszyna wirtualna, do której będą podłączać się użytkownicy.

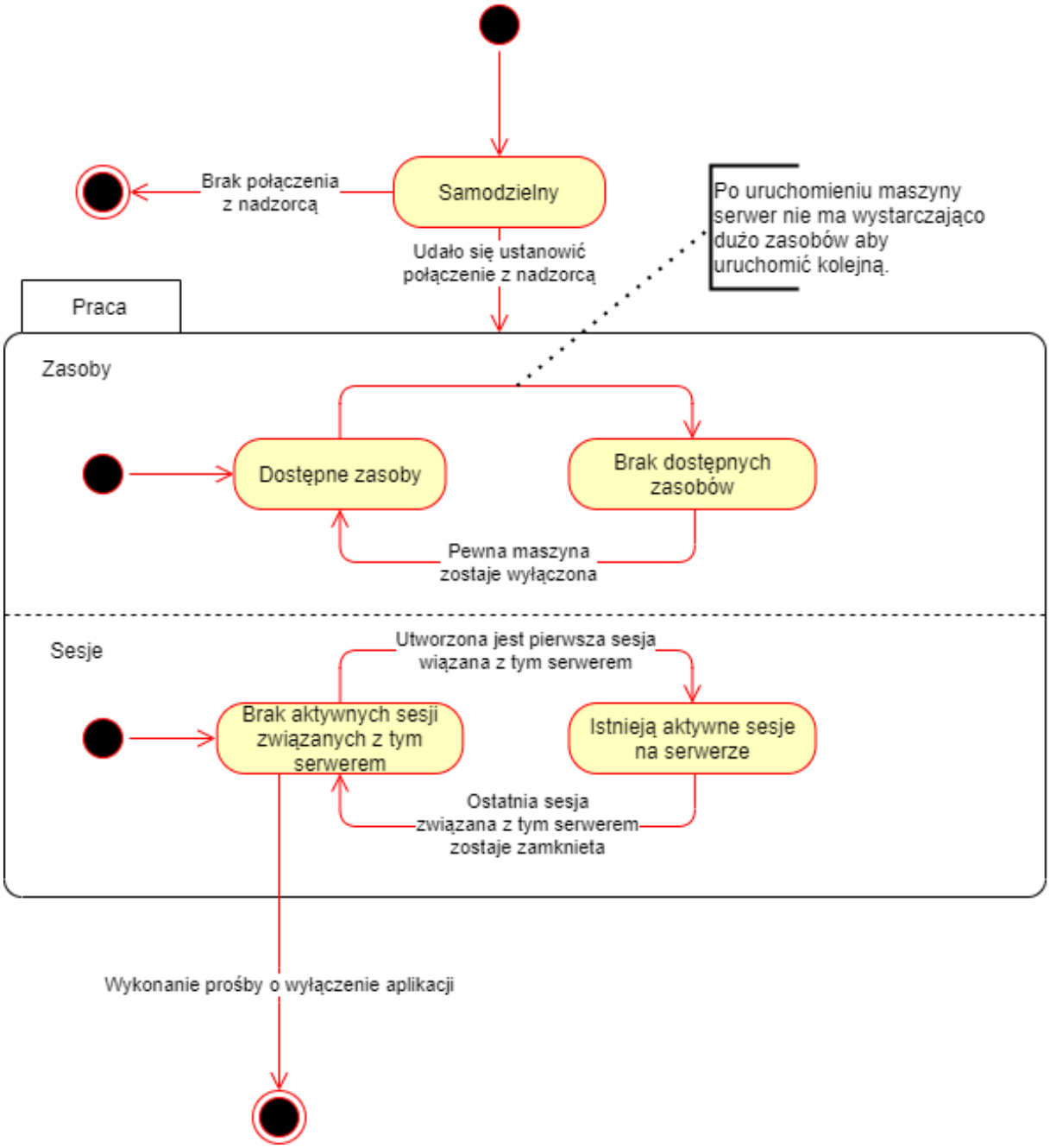


Rysunek 0.3: Diagram stanów dla maszyny wirtualnej

Maszyna, aby być całkowicie uruchomiona, musi zostać zaopatrzona we wszystkie konfiguracje. Stan "Wolnaóznacza możliwość przypisania sesji do tej maszyny. Po przypisaniu maszyny do sesji przechodzi ona w stan oczekiwania na użytkownika. Czas oczekiwania jest konfigurowalny, a po jego upłynięciu maszyna przechodzi w stan oczekiwania na wyłączenie. W tym stanie oczekuje ona na ponowne połączenie, pozwalając użytkownikowi na bezproblemowy powrót do sesji w przypadku nieoczekiwanego utracenia połączenia. Jeżeli użytkownik nie powróci do sesji, system wyłącza maszynę. Maszyna jest w stanie Źajęta", gdy aktualnie pracuje na niej użytkownik. Monitorowanie zajętości realizowane jest przy użyciu odpowiedniej kolejki wiadomości.)

0.3.2. Serwer wirtualizacji

Serwer wirtualizacji monitoruje zasoby zużywane przez uruchamiane na nim maszyny wirtualne oraz fakt podłączenia do niego użytkowników.



Rysunek 0.4: Diagram stanów dla serwera wirtualizacji

Przy starcie serwer wirtualizacji oczekuje na działającego nadzorcy w sieci. W przypadku jego braku serwer kończy działanie zwracając błąd. Pod względem zasobów, może on mieć wolne zasoby aby utworzyć nowe maszyny, lub też nie. Jednak ważniejszym stanem z perspektywy działania serwera są podłączeni do niego użytkownicy. W przypadku gdy podłączony jest do niego

przynajmniej jeden użytkownik, serwer nie może się poprawnie zakończyć pracy aż użytkownik nie skończy używać maszyny.

0.3.3. Użytkownik

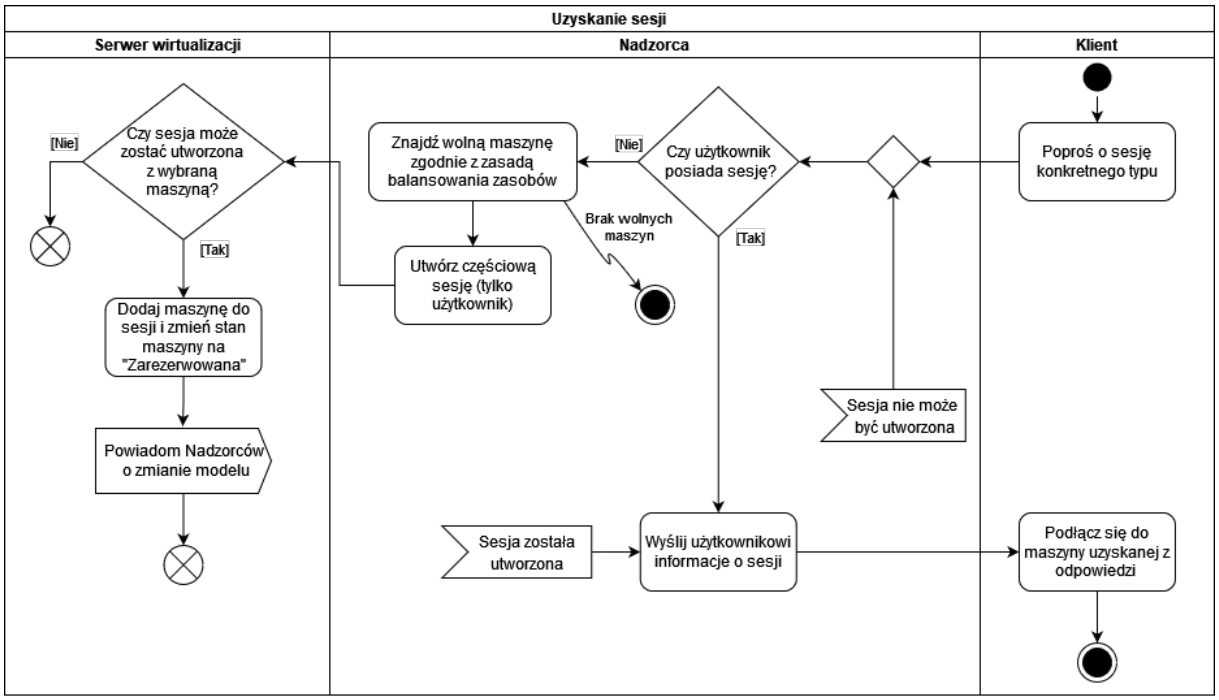
Rysunek 0.5: Diagram stanów dla użytkownika

Użytkownik z perspektywy systemu po zalogowaniu może być 2 dwóch stanach: pracuje w ramach swojej sesji lub też nie. W stanie "Połączonyąplikacja kliencka powiadamia serwer wirtualizacji, że ciągle jest obecny. Przy zmianie stanu do innego informowanie musi ustać, aby serwer mógł wykryć odłączenie się użytkownika.

0.4. Procesy biznesowe

0.4.1. Uzyskanie sesji

Proces opisuje prośbę klienta o ustanowienie dla niego sesji. Sesja może już istnieć lub zostać utworzona.



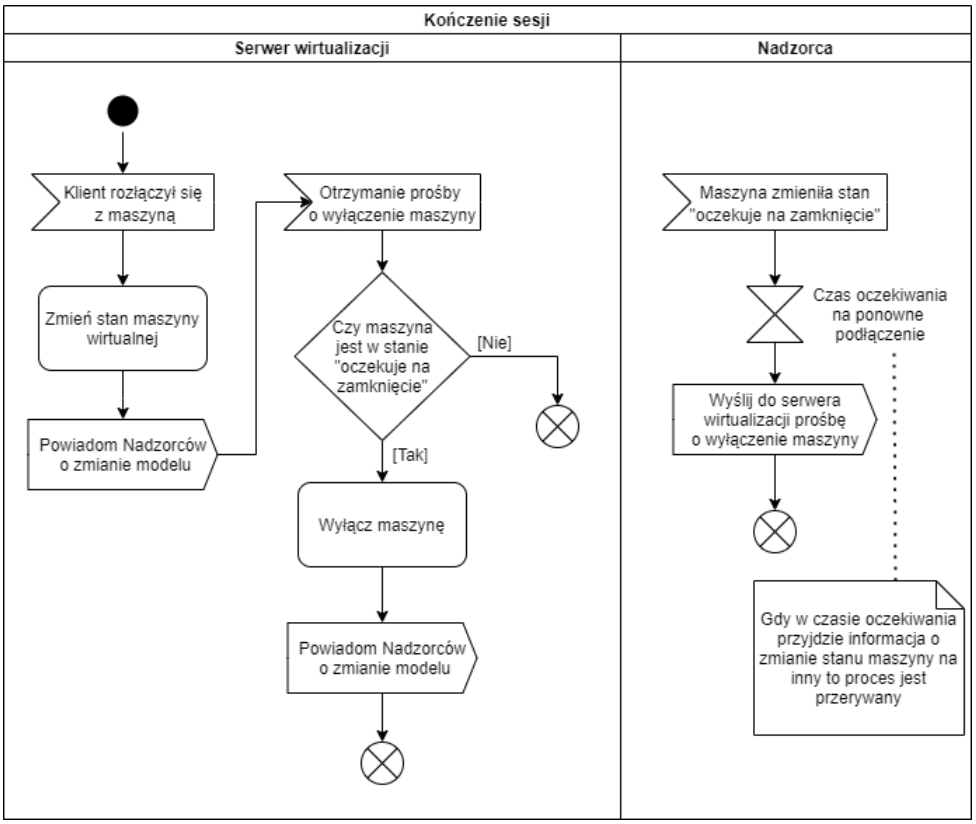
Rysunek 0.6: Diagram aktywności dla uzyskania sesji

W przypadku gdy sesja już istnieje zostaje one zwrócona użytkownikowi. W przeciwnym razie nadzorca, na podstawie modelu systemu, wybiera pewną wolna maszynę i wysyła do serwera wirtualizacji, na którym działa wybrana maszyna, prośbę o utworzenie sesji. Możliwość działanie wielu nadzorców wymaga, aby proces ten był powtarzalny, czyli dla konkretnego stanu modelu wybrana musi zostać ta sama maszyna. Gdy nie znajdzie wolnej maszyny, to zgłasza błąd do użytkownika. Z założenia taka sytuacja może zajść jedynie, gdy wszystkie maszyny zostały zajęte i brakuje zasobów na utworzenie nowych. Wynika to z tego, że system zawsze powinien trzymać pewien zapas wolnych maszyn. Może się zdarzyć, że model jest nieaktualny i nie można utworzyć sesji z wcześniej wybraną maszyną. Taka prośba zostaje odrzucona przez serwer wirtualizacji, ale zmiana modelu w nadzorcy, wywołana odświeżeniem modelu, spowoduje powtórzenie procesu, tym razem wybierając inną maszynę.

Uzyskanie sesji przez użytkownika zrealizowane jest asynchronicznie. Użytkownik oddzielnym zapytaniem prosi o uzyskanie sesji, po czym używając otrzymanego identyfikatora sesji prosi o jej dane. Obiekt jest w pełni utworzona, gdy odpowiedź nadzorcy sesję w stanie gotowym oraz adres maszyny do połączenia.

0.4.2. Kończenie sesji

Proces ma za zadanie zakończyć sesję oraz wyłączyć skojarzoną z nią wirtualna maszynę w celu zwolnienia zasobów.

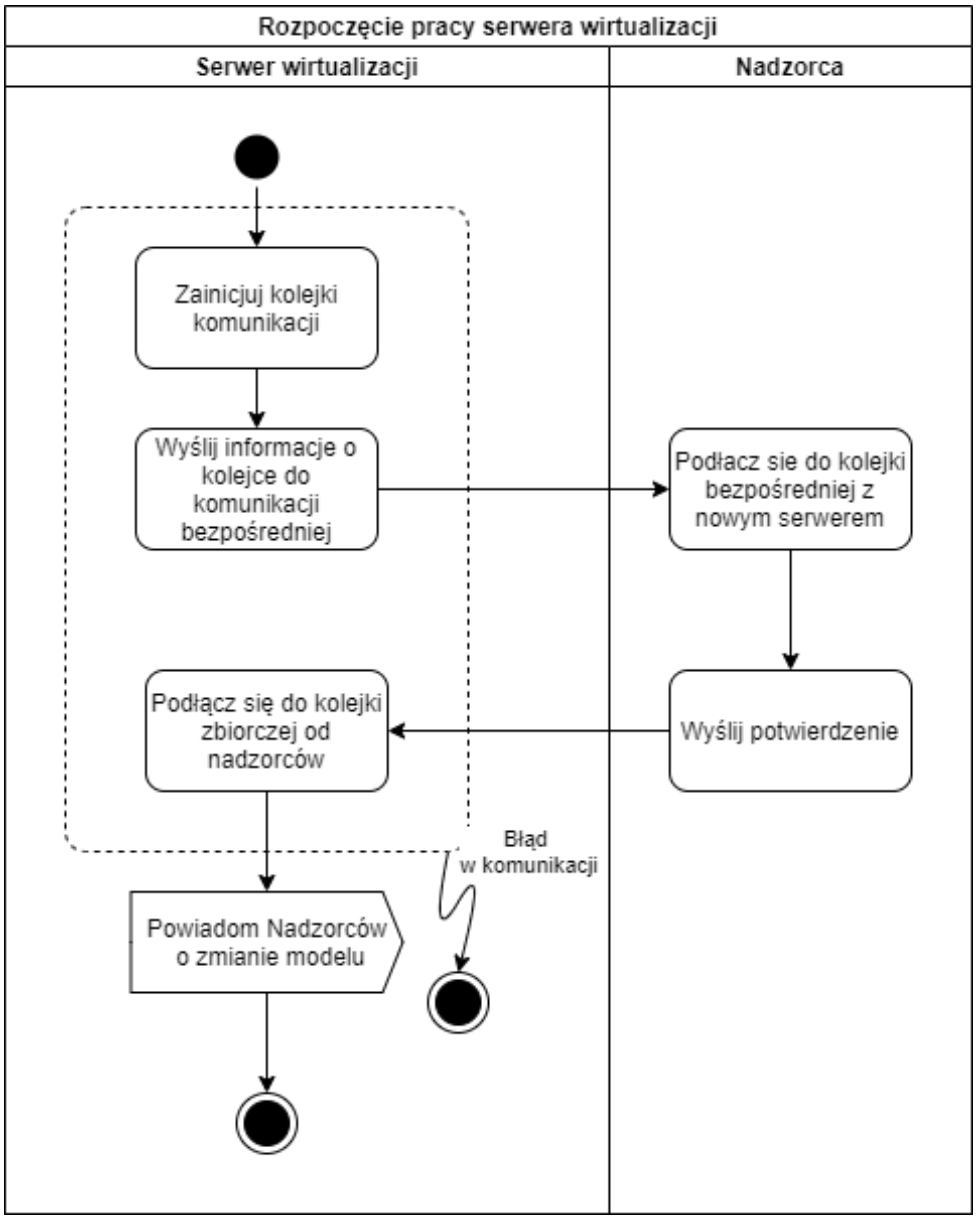


Rysunek 0.7: Diagram aktywności dla zakończenia sesji

Proces rozpoczyna się w momencie, gdy użytkownik odłączy się od systemu lub utraci połączenie. Po upłynięciu ustalonego czasu, jeżeli użytkownik nie podłączył się ponownie, maszyna zostaje wyłączona. serwer wirtualizacji informuję nadzorcę o utracie połączenia lub odłączeniu się użytkownika poprzez zmianę model. Decyzję o wyłączeniu maszyny podejmuje nadzorca. Powoduje to, że zmiana konfiguracji nadzorców będzie oznaczać spójną reakcje całego systemu. Dodatkowo umożliwia to w perspektywie czasu utworzenie bardziej złożonego algorytmu zarządzania zasobami.

0.4.3. Rozpoczęcie pracy serwera wirtualizacji

Proces opisuje przyjęcie nowego serwera wirtualizacji do systemu.



Rysunek 0.8: Diagram aktywności dla rozpoczęcia pracy serwera wirtualizacji

Serwer wirtualizacji bez działającego nadzorcy nie jest w stanie obsługiwać użytkowników. Oznacza to, że jeśli przy starcie nie wykryje brokera wiadomości lub nadzorcy po drugiej stronie kolejek[2] to się wyłączy. Jeżeli jednak komunikacja z nadzorcą jest możliwa, to serwer podłączy się do wspólnych kolejek oraz przekaże informacje nadzorcom o kolejce bezpośredniej. Gdy komunikacja będzie ustanowiona bezwarunkowo wyśle stan swojego modelu do nadzorców.

[2]Mechanizm potwierdzenia wykonania zadania

0.5. Komunikacja

0.5.1. Komunikacja wewnętrzna

Komunikacja wewnątrz systemu opiera się na kolejkach opisanych w opisie modułów. W celu uniknięcia wyścigów i utrzymania spójności modelu systemu pomiędzy nadzorcami ustalone są następujące zasady:

- Nadzorca może zmienić stan systemu jedynie w reakcji na odpowiedź serwera wirtualizacji. Odpowiedzi te wysyłane są do wszystkich nadzorców, dzięki czemu każdy nadzorca ma taki sam model systemu.

- Wiadomości przetwarzane są przez serwer wirtualizacji w sposób atomowy. Pojedyncza wiadomość musi zostać w pełni obsłużona zanim program przejdzie do obsługi kolejnej.

- Serwer wirtualizacji odpowiada na wiadomości wysyłając nowy stan maszyn. Jeżeli żądanie nie może być spełnione z powodu błędnego żądania, to serwer nie odpowiada na żądanie. Wyjątkiem jest żądanie o wysłanie aktualnego stanu maszyn.

- Z powodu asynchroniczności wiadomości moduły nie oczekują na odpowiedź. W przypadku nadzorcy przetwarzanie ódpowiedziźostanie uruchomione przez zmianę modelu.

- Do monitorowania utrzymania połączenia z brokerem użyty jest mechanizm zwracania wiadomości, które nie mogą zostać dostarczone[3]. Używając tego nadzorcy mogą wykryć, kiedy poszczególne serwery wirtualizacji przestaną działać, a serwery wirtualizacji - kiedy wszyscy nadzorcy przestaną działać.

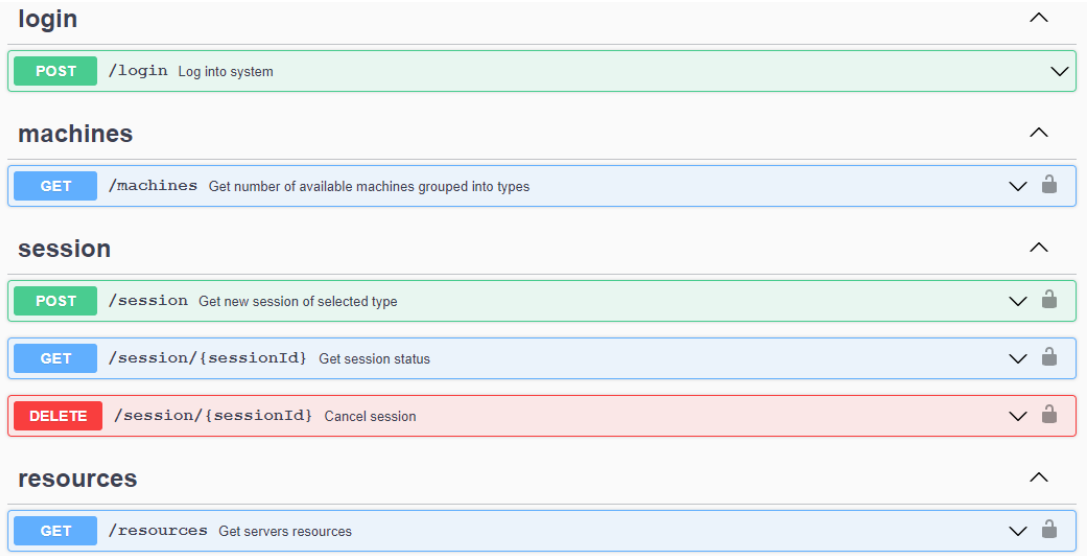
Opisane wyżej założenia pozwalają uniknąć problemu hazardów i wyścigów. Jeżeli wiele nadzorców wyśle do serwera wirtualizacji tą samą prośbę, np. o stworzenie sesji na konkretnej maszynie, to z atomowości obsługi sesja zostanie stworzona tylko dla pierwszego z nich. Serwer wirtualizacji wyśle wiadomość o aktualizacji stanu maszyn i zignoruje pozostałe prośby. Nadzorcy otrzymają zmianę stanów, co spowoduje wywołanie odpowiednich procedur. Dla pierwszego będzie to dalsza część procesu tworzenia sesji, a pozostali nadzorcy pozostaną w procesie wyszukiwania maszyny do sesji.

[3] Mechanizm zwracania wiadomości

0.5.2. Komunikacja zewnętrzna

Komunikacja aplikacji klienckiej oraz panelu administratora z systemem - nadzorcą - rozwiązana jest za pomocą REST API[4]. W zależności od konfiguracji nadzorcy wiadomości mogą być wysyłane za pomocą protokołu HTTPS[5], który zapewnia ich szyfrowanie. W tym celu wymagane jest, aby na adres, pod którym udostępniony będzie system, wystawiony był odpowiedni certyfikat[6], gwarantujący jego tożsamość.

Całość specyfikacji API umieszczona jest w załączniku. Poniżej znajduje się zestawienie oraz krótki opis endpointów.



Rysunek 0.9: Endpointy API

- Login - służy do logowania do systemu; współdzielony przez aplikację kliencką oraz panel administracyjny. Poprawne zalogowanie zwraca token do dalszej autoryzacji.
- Machines - służy do pobierania przez aplikację informacji o typach i ilości dostępnych maszyn. Ten endpoint, oraz wszystkie następne wymagają autoryzacji poprzez umieszczenie tokenu otrzymanego podczas logowania w odpowiednim nagłówku wiadomości, oraz dostępne są tylko dla użytkownika.
- Session - pozwala na wysłanie prośby o uzyskanie sesji, pobranie stanu sesji oraz jej anulowanie. Utworzenie sesji jest możliwe poprzez `POST` z typem maszyny. W odpowiedzi użytkownik dostaje częściowo wypełniony obiekt sesji zawierający id umożliwiające dalsze zapytania. `GET` zwraca obiekt sesji z aktualnym stanem. Jeżeli sesja jest gotowa, to zawiera on

[4]Opis REST API
[5]Specyfikacja protokołu HTTP Over TLS
[6]Opis certyfikatu TLS/SSL

też adres, z którym należy nawiązać połączenie RDP. `DELETE` umożliwia anulowanie sesji.

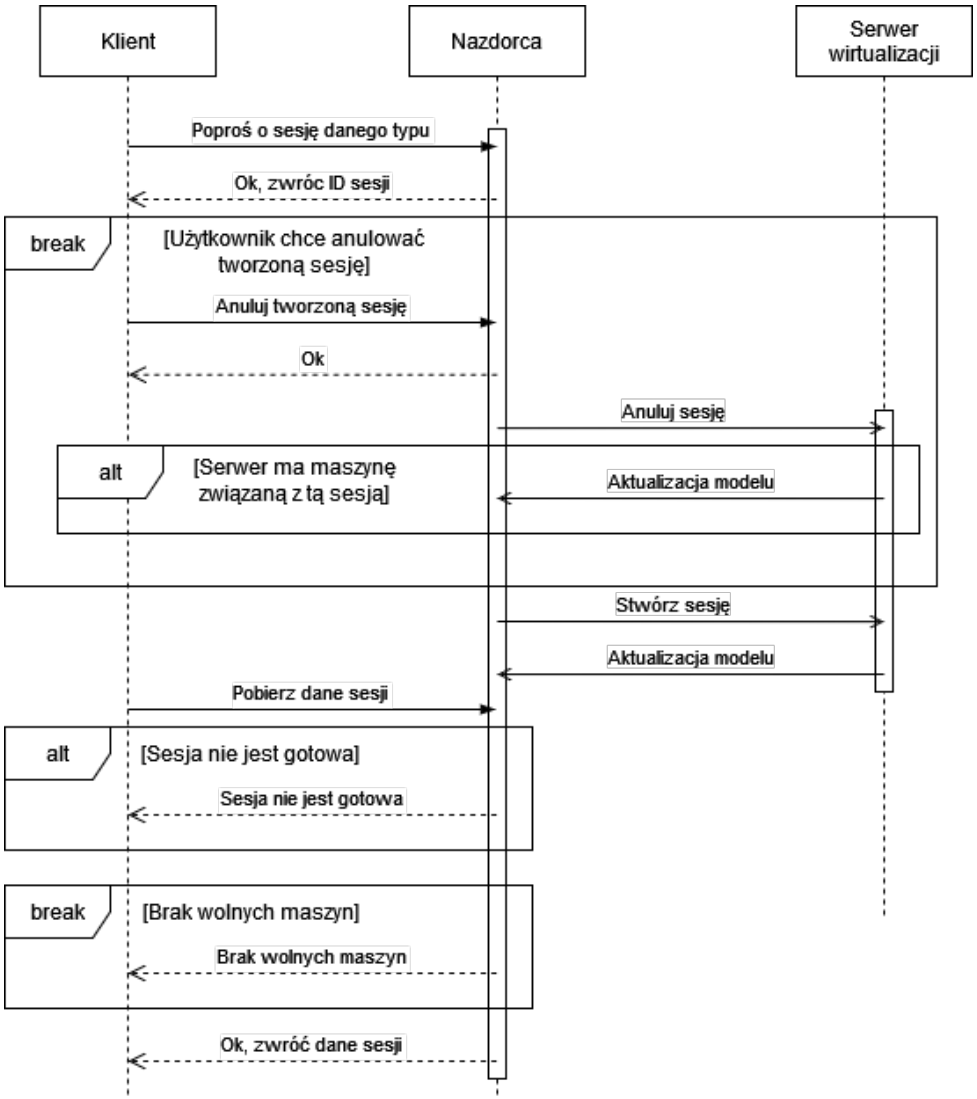
- Resources - udostępnia informację o zasobach działających serwerów wirtualizacji. Dostępny jedynie dla administratora.

Ważną informacją jaką musi posiadać system to fakt, czy użytkownik rzeczywiście jest podłączony do maszyny wirtualnej. System uzyskuje tą informację komunikując się z brokerem wiadomości odpowiedzialnym za komunikację z użytkownikami. Każda aplikacja kliencka po podłączeniu się do maszyny wirtualnej poprzez protokół RDP tworzy kolejkę o takiej nazwie jak uzyskany identyfikator sesji. Serwer wirtualizacji sprawdza co jakiś czas, czy na końcu kolejki istnieje jakikolwiek konsument. Gdy użytkownik się rozłączy to kolejka jest usuwana przez aplikację kliencką, co pozwala serwerowi wykryć odłączenie się użytkownika.

0.6. Sekwencje komunikacji

0.6.1. Utworzenie sesji

Celem tej sekwencji komunikacji jest odnalezienie istniejącej już sesji lub stworzenie nowej. Zakładamy, że w systemie zawsze jest jakaś wolna maszyna. Inaczej zgłaszamy użytkownikowi błąd.



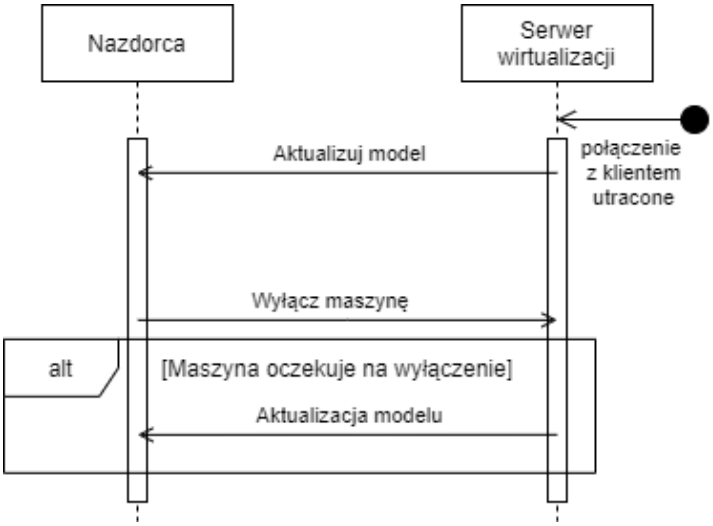
Rysunek 0.10: Sekwencja komunikacji utworzenia sesji

Po prośbie użytkownika nadzorca znajduje wolna maszynę i prosi konkretny serwer wirtualizacji aby spróbował utworzyć sesje dla pewnego użytkownika. Gdy to się uda, ten wysyła zbiorczą kolejką do nadzorców informacje o zmianie modelu. W przeciwnym przypadku nadzorca powtarza wyszukanie wolnej maszyny. O tym, czy nadzorca musi powtórzyć wyszukiwanie decyduje stan otrzymanej maszyny (m.in. czy sesja do niej przypisana należy do tego użytkownika).

Może się zdarzyć także anulowanie wyszukiwania przez użytkownika. Wtedy jeżeli maszyna jest już przydzielona użytkownikowi, to serwer wirtualizacji jest powiadamiany o anulowaniu sesji, aby ją anulował. Jeżeli nie została jeszcze utworzona, to nadzorca dopilnuje aby więcej nie szukać sesji lub wyłączy ją w miarę potrzeby.

0.6.2. Zakończenie sesji

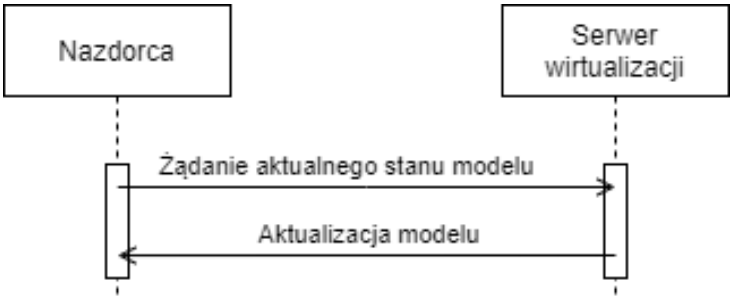
Sekwencja ta zainicjowana jest poprzez utracenie połączenia z użytkownikiem maszyny. Serwer powiadamia o tym fakcie nadzorców, po czym oczekuje na polecenie wyłączenia maszyny przesłane przez nadzorcę. Serwer może odmówić z powodów różnic modelu, lub jeżeli maszyna znów jest używana przez użytkownika, ignorując wiadomość.



Rysunek 0.11: Sekwencja komunikacji zakończenia sesji

0.6.3. Aktualizacja stanu

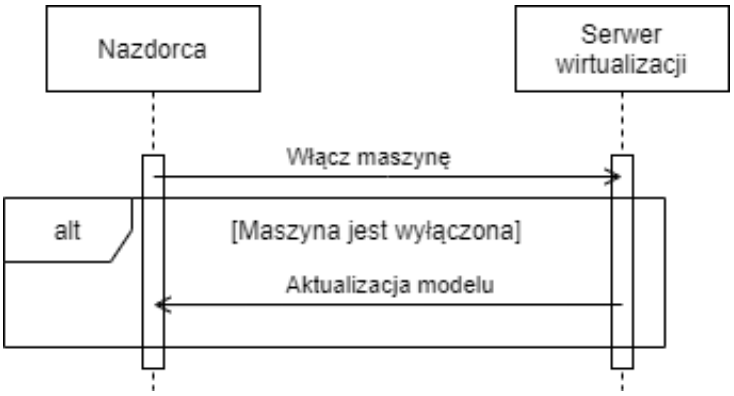
Nadzorca może w każdej chwili poprosić wszystkie serwery wirtualizacji o przesłanie ich aktualnego stanu poprzez wspólna kolejkę do serwerów wirtualizacji. Serwery muszą bezwarunkowo odpowiedzieć aktualnym stanem do wspólnej kolejki zwrotnej.



Rysunek 0.12: Sekwencja komunikacji aktualizacji stanu systemu

0.6.4. Włączenie maszyny

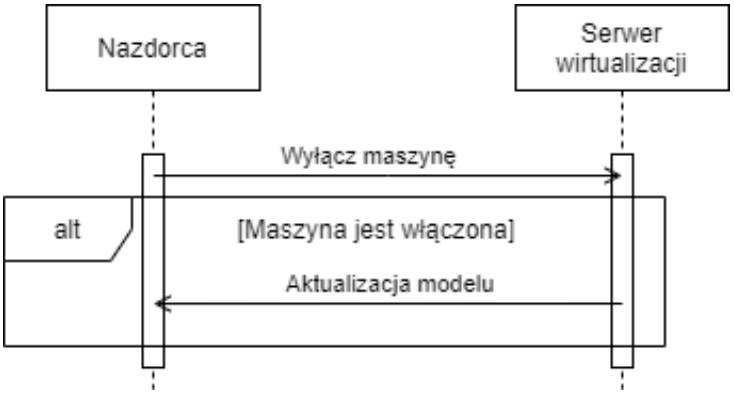
Nadzorca może poprosić konkretny serwer wirtualizacji aby utworzył maszynę o konkretnej nazwie. Jeżeli maszynie nie istnieje, to zostanie uruchomiona oraz serwer odeśle powiadomienie zbiorcza kolejką o zmianie modelu. W przeciwnym wypadku nie zrobi nic.



Rysunek 0.13: Sekwencja komunikacji włączenia maszyny

0.6.5. Wyłączenie maszyny

Nadzorca może poprosić konkretny serwer wirtualizacji aby wyłączył konkretną maszynę wirtualną. Jeżeli maszynę można wyłączyć to zostanie on wyłączona. Następnie serwer wirtualizacji odeśle powiadomienie zbiorcza kolejką o zmianie modelu. W przeciwnym wypadku nie zrobi nic.



Rysunek 0.14: Sekwencja komunikacji wyłączenia maszyny