**Felipe Arado**

**Gleberson Inácio**

**Rafael Fernando Cotrim**

**Vitor Piassi**

Elektra

**Centro Universitário Toledo**

**Araçatuba**

**2015**

**Felipe**

**Gleberson Inácio**

**Rafael Fernando Cotrim**

**Vitor Piassi**

Elektra

Trabalho de conclusão de semestre apresentado para obtenção da nota final da disciplina de projeto do curso de Análise e Desenvolvimento de Sistemas do Centro Universitário Toledo sob a orientação do Professor Evandro Cesar Estevam.

**Centro Universitário Toledo**

**Araçatuba**

**2015**

Resumo

O documento em questão tem, como objetivo, demonstrar as funcionalidades de um aplicativo para dispositivos móveis, capaz de automatizar uma residência e permitir ao usuário o controle dos componentes presentes nos cômodos da casa.

Além do aplicativo, o sistema Elektra conta com a plataforma de prototipagem eletrônica Arduino, seus componentes (Leds, Sensores, etc) e propõe aos usuários uma facilidade maior ao realizar tarefas diárias dentro da residência.

Palavras-chave: desenvolvimento, automação, residencia.

Abstract

The document in question has the objective to demonstrate the functionality of an application for mobile devices, able to automate a residence and allow the user control of the components present in the rooms of the house.

In addition to the application, the system has Elektra electronics prototyping platform Arduino and its components (LEDs, sensors, etc.) and offers users a greater ease to perform daily tasks inside the residence.

Keywords: development, automation, residence.

Sumário

[1. DOCUMENTO DE REQUISITOS 5](#_Toc423445363)

[1.1. Visão geral do sistema 5](#_Toc423445364)

[1.2. Problemas existentes 5](#_Toc423445365)

[1.3. Escopo do sistema 5](#_Toc423445366)

[1.3.1. Requisitos funcionais 5](#_Toc423445367)

[1.3.2. Requisitos não funcionais 5](#_Toc423445368)

[2. ANÁLISE E MODELAGEM DE REQUISITOS 6](#_Toc423445369)

[2.1. Lista de casos de us 6](#_Toc423445370)

[2.2. Modelos de casos de uso 7](#_Toc423445371)

[2.3. Modelo de Classe do Domínio do Negócio 8](#_Toc423445372)

[2.4. Detalhamento dos Casos de Uso e Diagrama de Sequência 9](#_Toc423445373)

[3. PROJETO DO SISTEMA 23](#_Toc423445374)

[3.1. Ambiente de Desenvolvimento 23](#_Toc423445375)

[3.2. Modelo conceitual de banco de dados 23](#_Toc423445376)

[3.2.1 Modelo Conceito de banco de dados (Web Service) 23](#_Toc423445377)

[3.2.2 Modelo Conceito de banco de dados (Android) 24](#_Toc423445378)

[3.4. Modelo Físico de Banco de Dados (Tablas e Scripts) 25](#_Toc423445379)

[3.4.1 Script do Web Service 25](#_Toc423445380)

[3.4.2 Script do Android 27](#_Toc423445381)

[3.5. Modelos de Interface 29](#_Toc423445382)

[3.6. Arquitetura de Classe 36](#_Toc423445383)

[3.7. Modelo de Sequencia 36](#_Toc423445384)

[3.8. Modelo de Componente 36](#_Toc423445385)

[4. QUALIDADE E SEGURANÇA DE SOFTWARE 37](#_Toc423445386)

[5. CONCLUSÃO 40](#_Toc423445387)

[6. REFERÊNCIAS BIBLIOGRÁFICAS 41](#_Toc423445388)

[7. CRONOGRAMA 42](#_Toc423445389)

1. DOCUMENTO DE REQUISITOS

O sistema proposto tem como objetivo, realizar uma automação residêncial, permitindo assim facilitar as tarefas diárias de uma residência.

* 1. Visão geral do sistema

O sistema Elektra tem como objetivo monitorar e controlar uma casa que pode acionar os equipamentos de iluminação, sensores, ventiladores e etc, através de um aplicativo android.

* 1. Problemas existentes

O objetivo do projeto é trazer facilidade e, consequentimente, melhorar a qualidade de vida, reduzindo o trabalho doméstico, aumentando o bem estar e a segurança de seus habitantes com redução do consumo de energia.

* 1. Escopo do sistema
     1. Requisitos funcionais

As funcionalidades disponíveis para os usuários da residência são de ligar e desligar os cômodos e sensores que auxiliam nos processos do sistema, tais como:

* Controle de Ambiente.
* Controle de Temperatura.
* Controle de Segurança(Alarme).
  + 1. Requisitos não funcionais

Os requisitos não funcionais serão empregados para a melhor usabilidade do sistema, levando em consideração os conceitos:

* Mensagens claras e objetivas
* Telas de fácil entendimento
* Design harmônico e direto.

1. ANÁLISE E MODELAGEM DE REQUISITOS
   1. Lista de casos de us

|  |  |  |  |
| --- | --- | --- | --- |
| **Nº** | **NOME DO CASO DE USO** | **DESCRIÇÃO** | **ATOR** |
| 1 | Efetuar Login | Efetuar Login No Sistema | Usuário |
| 2 | Ligar Luz | Permite Ligar A Luz | Usuário |
| 3 | Desligar Luz | Permite Desligar A Luz | Usuário |
| 4 | Ativar Sensor Chuva | Permite Ativar O Sensor De Chuva | Usuário |
| 5 | Desativar Sensor Chuva | Permite Desativar O Sensor De Chuva | Usuário |
| 6 | Ligar Sensor De Luminosidade | Permite Ligar O Sensor De Luminosidade | Usuário |
| 7 | Desligar Sensor De Luminosidade | Permite Desligar O Sensor De Luminosidade | Usuário |
| 8 | Grava Log | Grava O Log No Sistema | Sistema |
| 9 | Atualiza Status | Atualiza O Status Do Sistema | Sistema |

* 1. Modelos de casos de uso

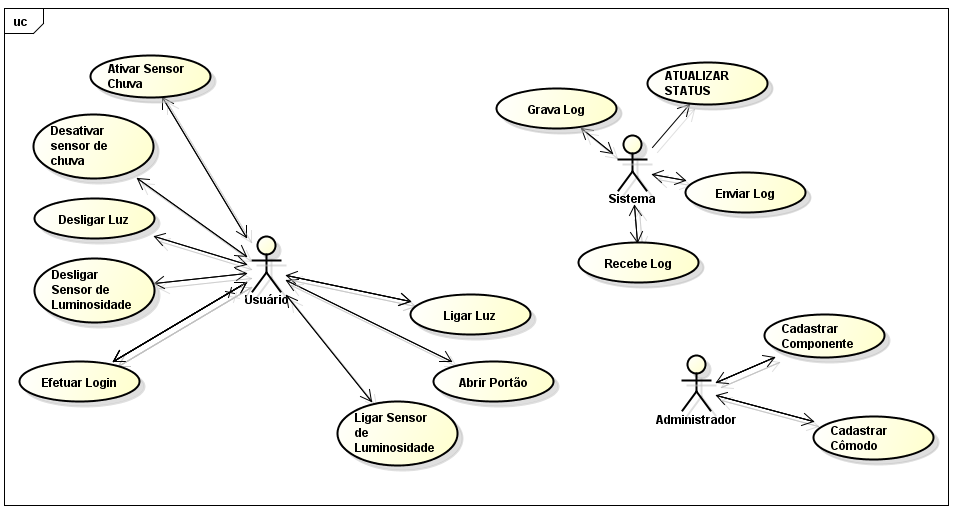


Figura 1 – Diagrama de Casos de Uso

* 1. Modelo de Classe do Domínio do Negócio

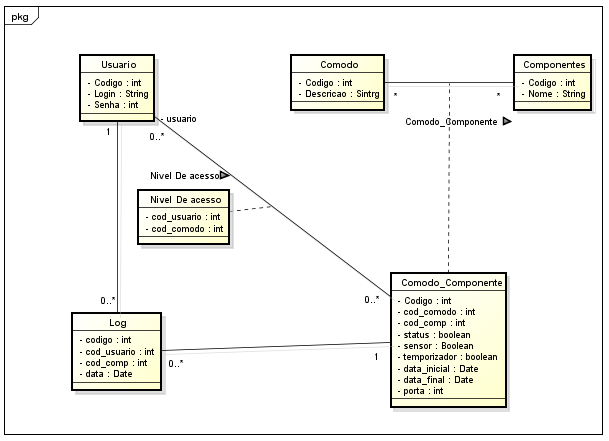
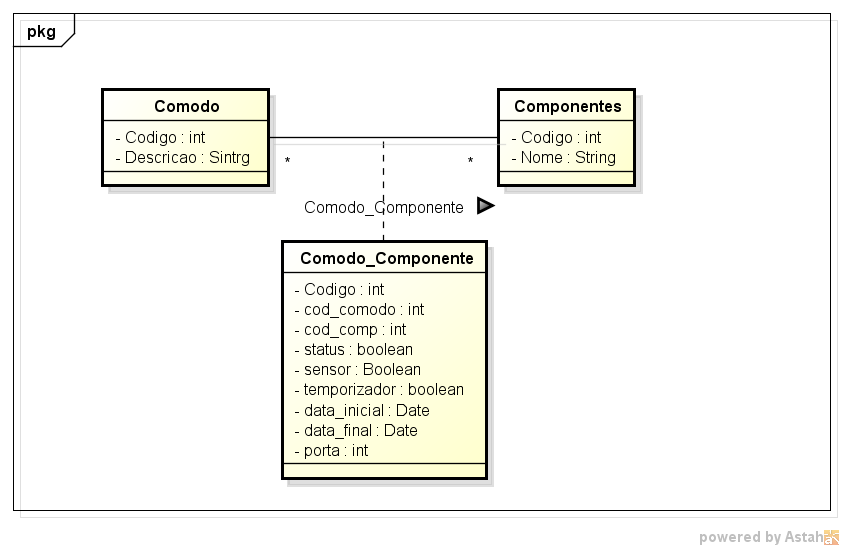


Figura 2 – Diagrama de Classes Web Service

  
Figura 3 – Diagrama de Classes Aplicação Androide

* 1. Detalhamento dos Casos de Uso e Diagrama de Sequência

**Caso de Uso 1** – Efetuar Login

**Objetivo:** Autenticar usuário para permitir o acesso ao sistema.

**Ator:** Usuário**.**

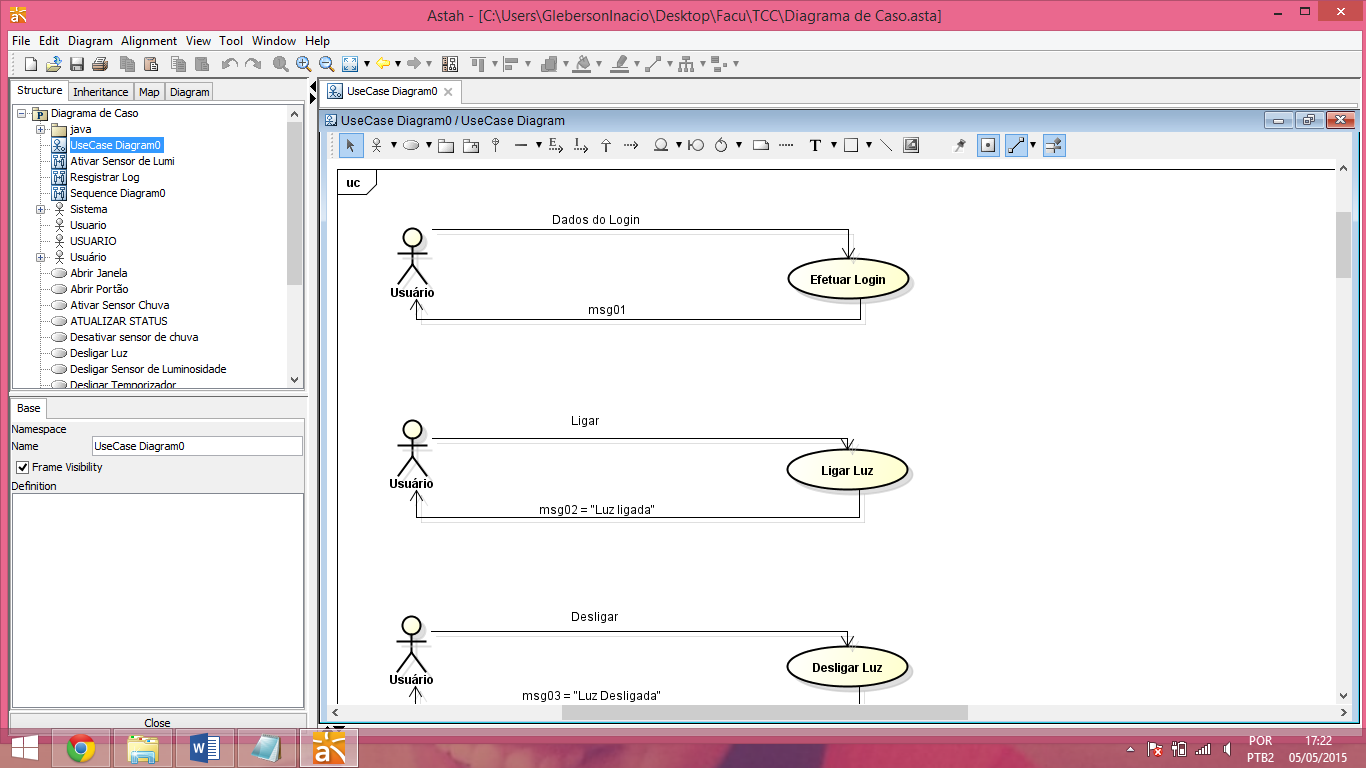


Figura 4 – Diagrama de Caso de Uso Efetuar Login

**Cenário Principal:**

1 - Usuário Abre o sistema.

2 - Usuário Digita o login e a senha.

3 - Usuário clica no botão / Sistema envia login e a senha pro webservice.

5 - Webservice service valida, atualiza o status dos componentes e devolve os dados necessários. /<<Include>> Grava Log.

6 – Sistema emite MSG01 e abre a tela inicial.

**Cenários Alternativo:**

Login e Senha Vazios

2.1 - Usuário não digita nada.

2.2 - Usuário clica no botão.

2.3 - Sistema envia login e a senha para o webservice.

2.4 - Webservice service valida e devolve a permissão. /

2.5 - Sistema abre a MSG01 - "Por favor digitar login e senha".

2.6 - Encerra o caso de uso e volta ao passo 2 do cenário principal.

Login ou Senha Inválido

3.1 - Usuário clica no botão / Sistema envia login e a senha pro webservice.

3.3 - Webservice service valida e devolve a permissão.

3.4 - Sistema abre a MSG01 - "Usuário ou senha inválidos".

3.5 - Encerra o caso de uso e volta ao passo 2 do cenário principal.

**Diagrama de Sequência:**

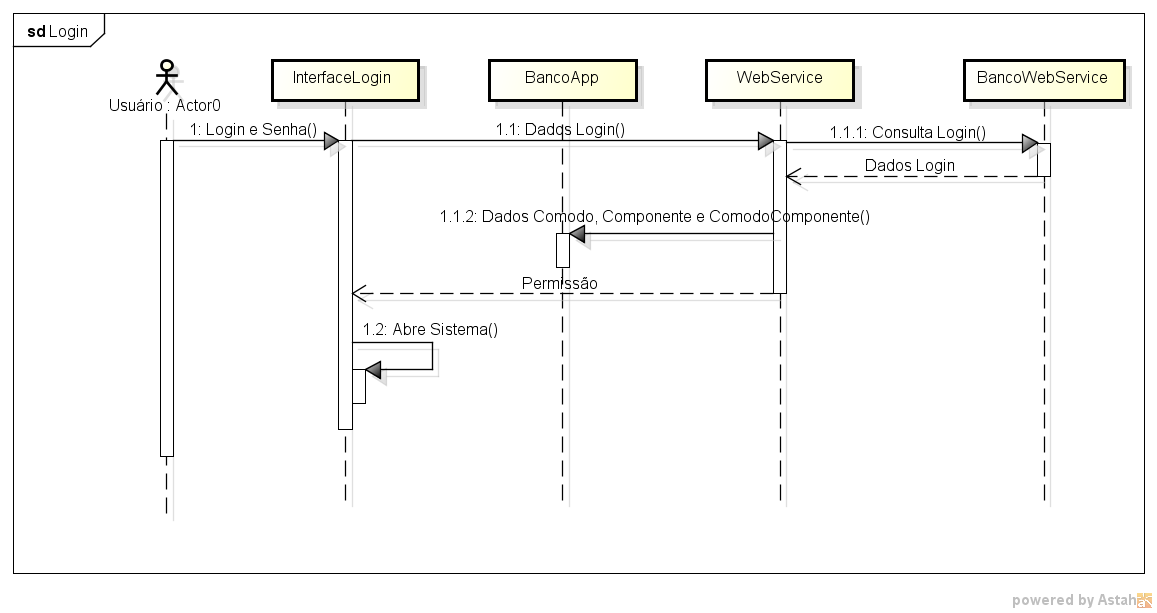


Figura 5 – Diagrama de Sequencia Efetuar Login

**Caso De Uso 2 -** Ligar a luz

**Objetivo:** Ligar a luz de um cômodo qualquer.

**Ator:** Usuário

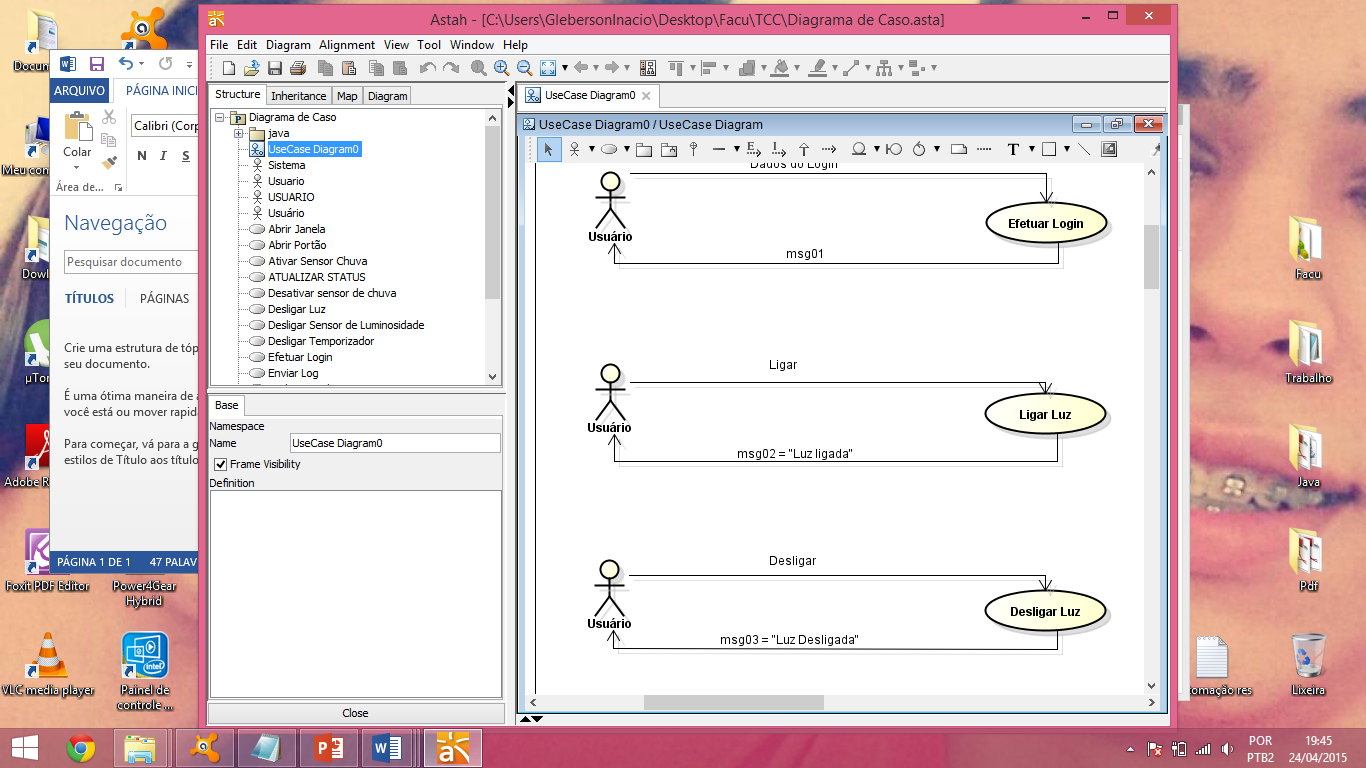


Figura 6 – Diagrama de Caso de Uso Ligar Luz

**Cenário principal:**

1. Usuário seleciona o cômodo desejado.
2. Usuário seleciona a opção Ligar luz.
3. Sistema envia os dados para o WebService
4. WebService Valida o status grava no banco e devolve ao sistema. /<<include>> Gravar log.
5. Sistema emite MSG02.

**Cenário Alternativo:**

Luz já ativada

4.1. WebService Valida que a luz já está ativada e retorna o status ao sistema.

4.2. Sistema emite MSG02 “A luz está ligada” e retorna ao passo 1.

**Diagrama de Sequência:**

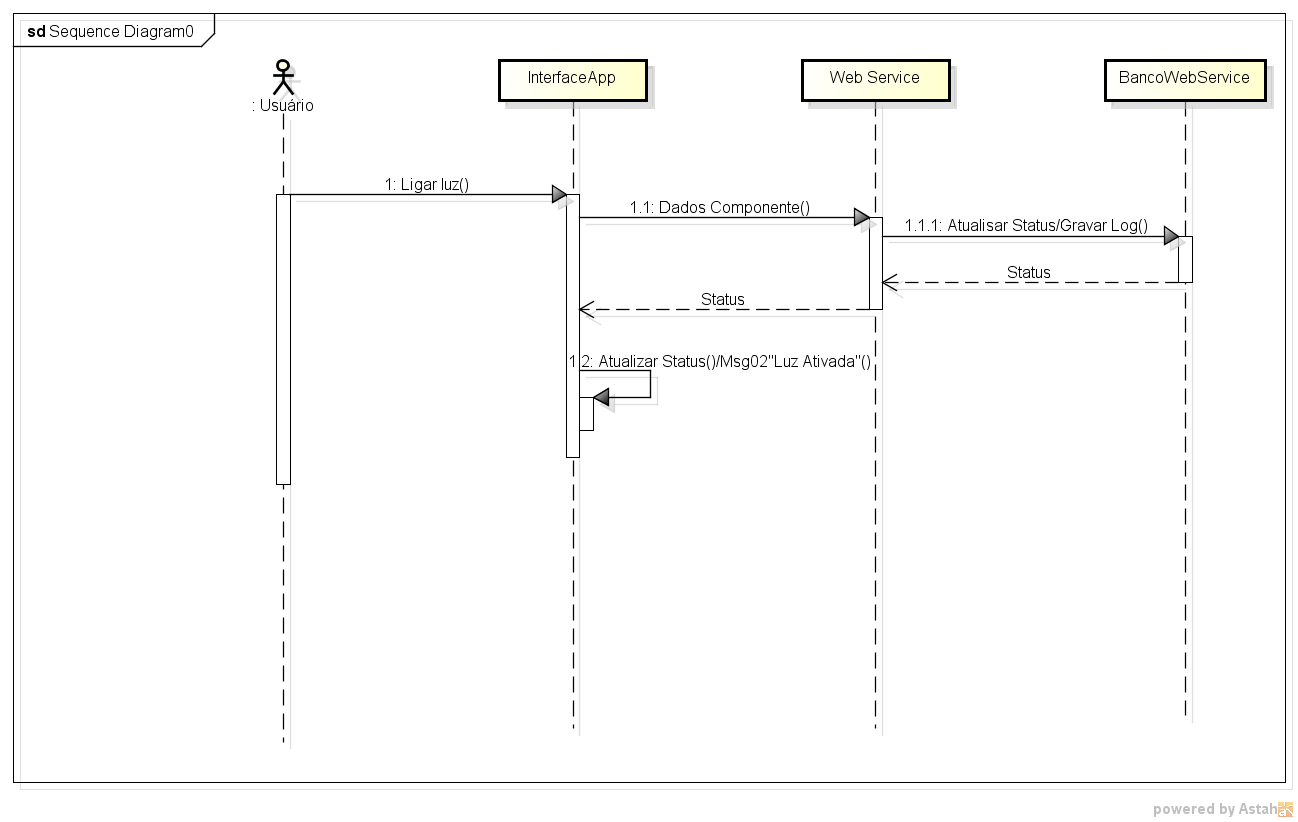


Figura 7 – Diagrama de Sequencia Ligar Luz

**Caso de Uso 3** – Desligar Luz

**Objetivo:** Desligar a luz.

**Ator:** Usuário

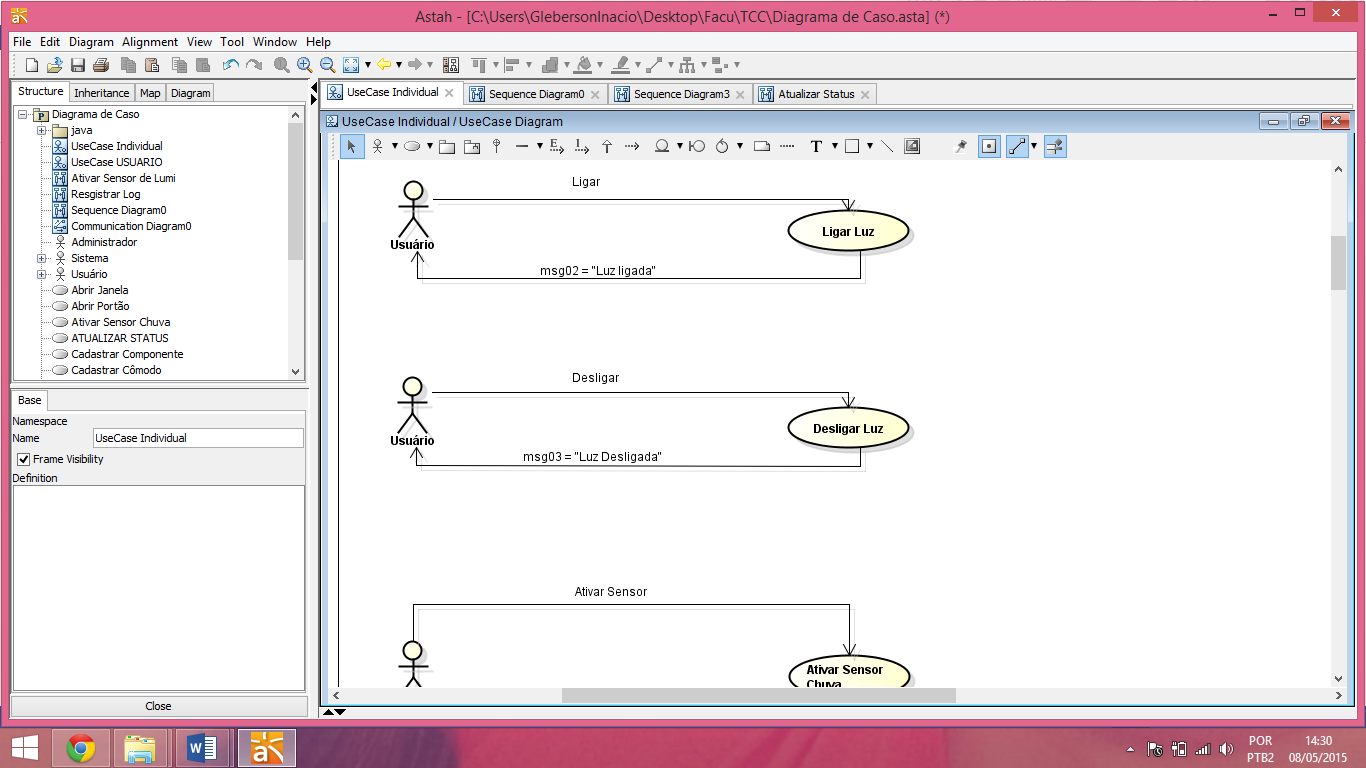


Figura 8 – Diagrama de Caso de Uso Desligar Luz

**Cenário principal:**

1. Usuário seleciona o cômodo desejado.
2. Usuário seleciona a opção desligar luz.
3. Sistema envia os dados para o WebService
4. WebService Valida o status grava no banco e devolve ao sistema. /<<include> gravar log.
5. Sistema emite MSG03.

**Cenário Alternativo:**

Luz já desativada

4.1. WebService Valida que a luz já está desativada e retorna o status ao sistema.

4.2. Sistema emite MSG03 “A luz está Desligada” e retorna ao passo 1.

**Diagrama de Sequência:**

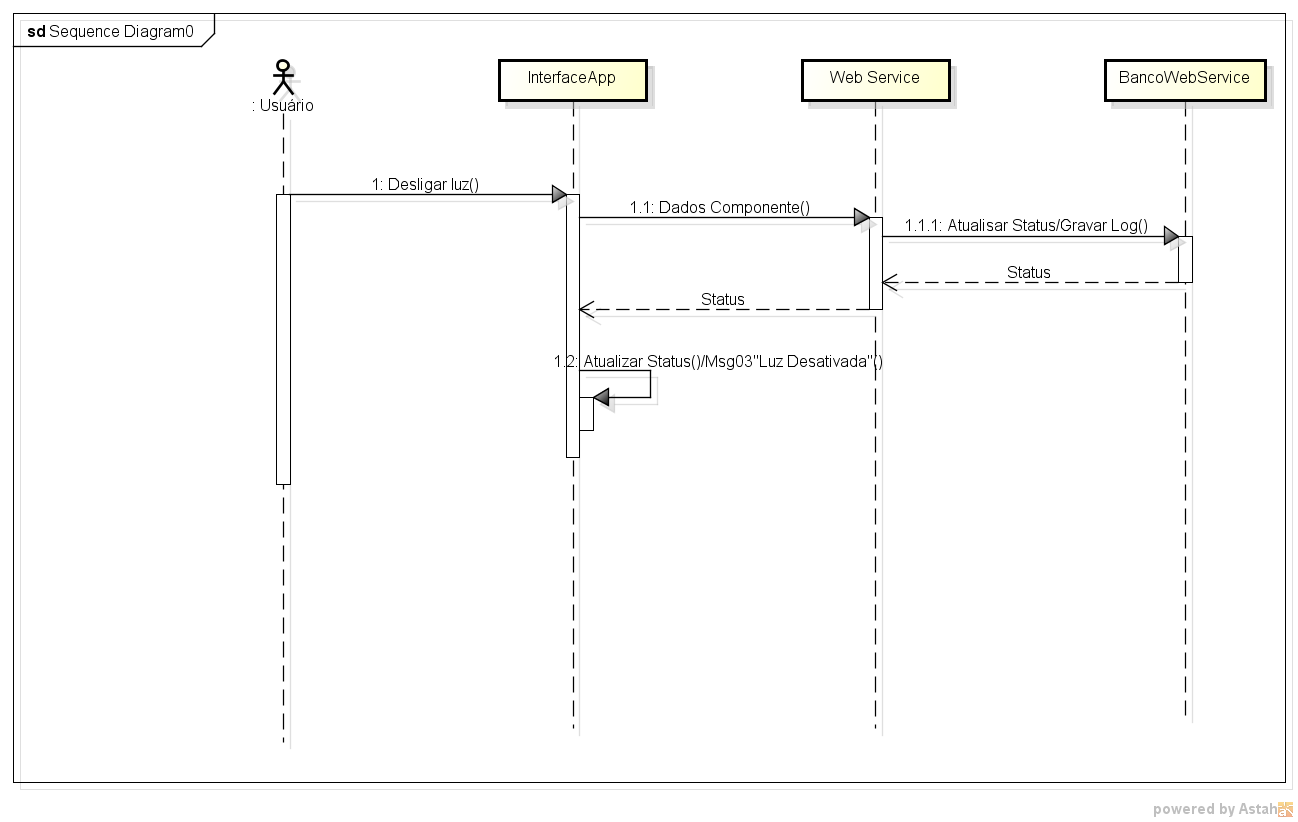


Figura 9 – Diagrama de Sequencia Desligar Luz

**Caso de Uso 4** – Ativar Sensor De Chuva

**Objetivo:** Ativar nos parâmetros o sensor de chuva.

**Ator:** Usuário.

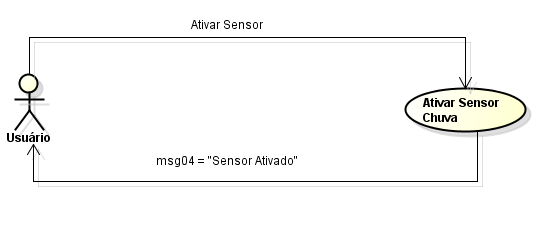


Figura 10 – Diagrama de Caso de Uso Ativar Sensor Chuva

**Cenário Principal**

1. O usuário abre os parâmetros para configuração.
2. Clica na opção para ativar o sensor.
3. O Sistema envia os parados para o WebService.
4. WebService valida o status grava no banco e devolve para o sistema / <<include>> Gravar Log.
5. Sistema Emite MSG04.

**Cenários Alternativos**

Caso o sensor já esteja ligado.

4.1 WebService Valida que o sensor já está ligado e retorna o status para o sistema.

4.2 Sistema emite MSG 04 “Sensor Já Ativado” e retorna ao passo 1.

**Diagrama de Sequência**

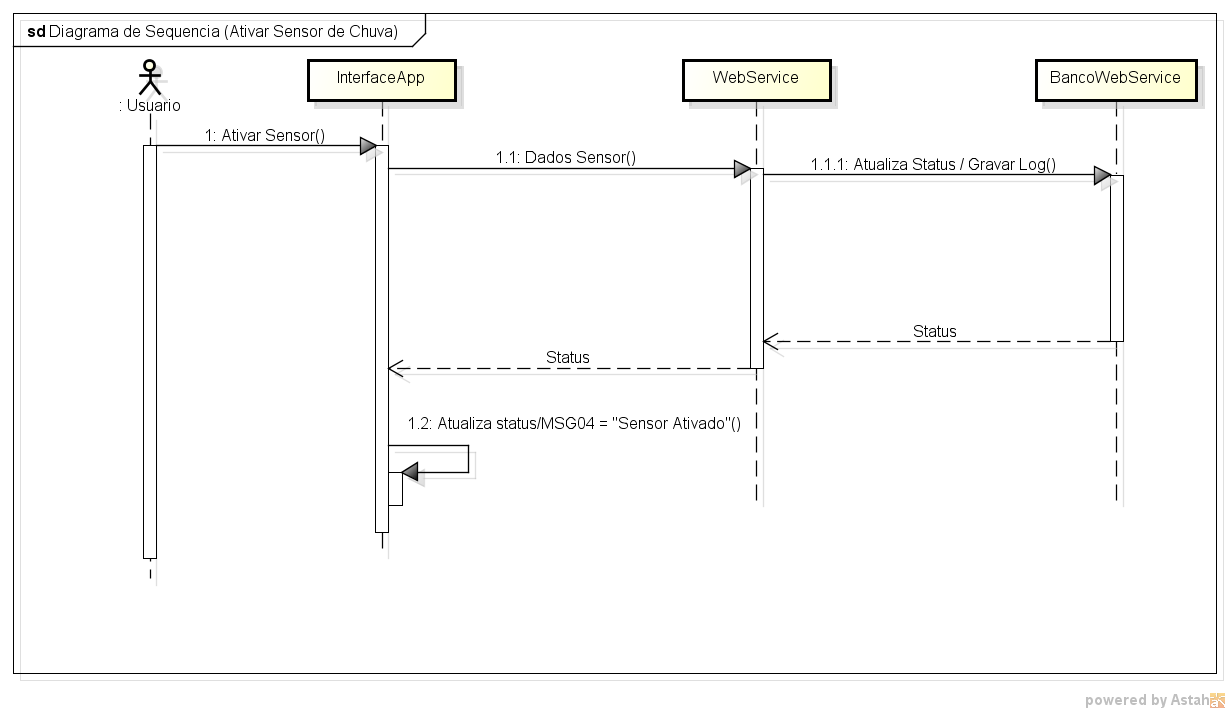


Figura 11 – Diagrama de Sequencia Ativar Sensor Chuva

**Caso De Uso 5** - Desativar sensor de chuva

**Objetivo:** Desativar o sensor de chuva

**Ator:** Usuário

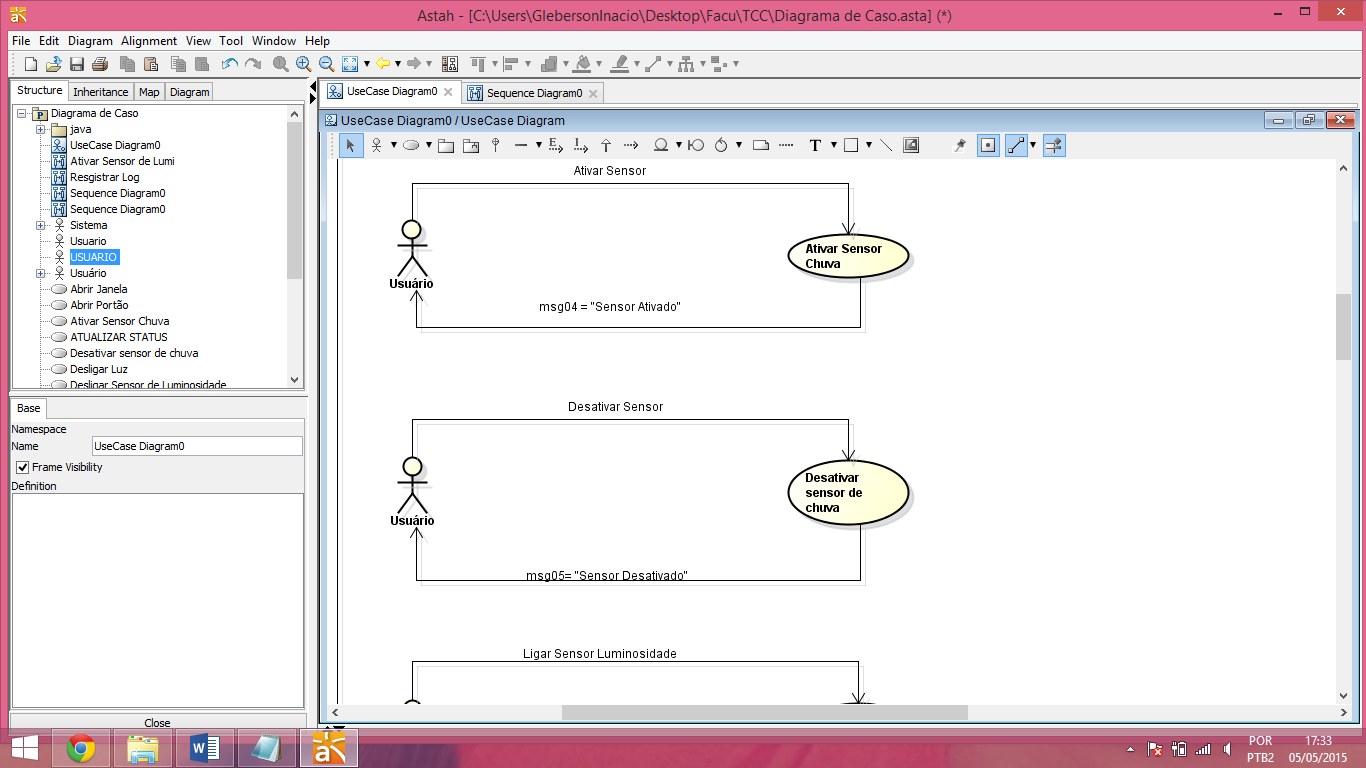


Figura 12 – Diagrama de Caso de Uso Desativar Sensor Chuva

**Cenário Principal:**

1. Usuário escolhe um cômodo.
2. Sistema traz os componentes.
3. Usuário escolhe desativar o sensor de chuva.
4. Manda para o webservice o status
5. Webservice valida e grava o status do sensor no banco, grava o log e devolve para o sistema. / <<Include>> Grava Log.
6. Sistema Emite MSG05.

**Cenário Alternativo:**

Sensor de chuva já desativado

* 1. Webservice valida e devolve a MSG05 - "Sensor já desligado".
     1. Encerra o caso de uso e volta ao passo 2 do cenário principal.

**Diagrama de Sequência:**

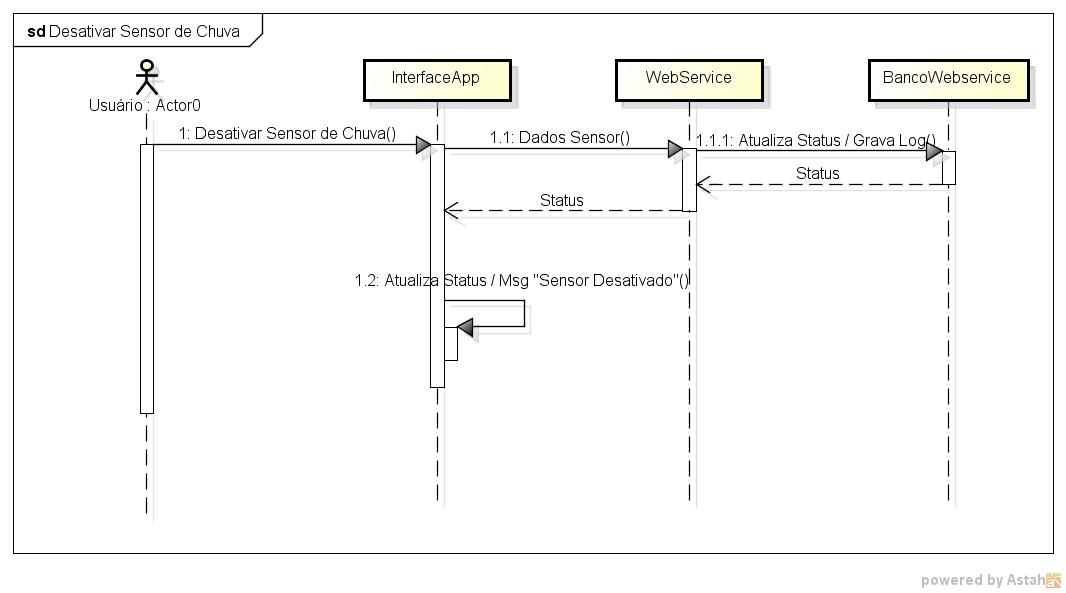


Figura 13 – Diagrama de Sequencia Desativar Sensor Chuva

**Caso de uso 6 -** Ligar Sensor De Luminosidade

**Objetivo:** Ativar sensor de luminosidade para que as luzes selecionadas pelo usuário acendam automaticamente quando não houver luz natural.

**Ator** – Usuário

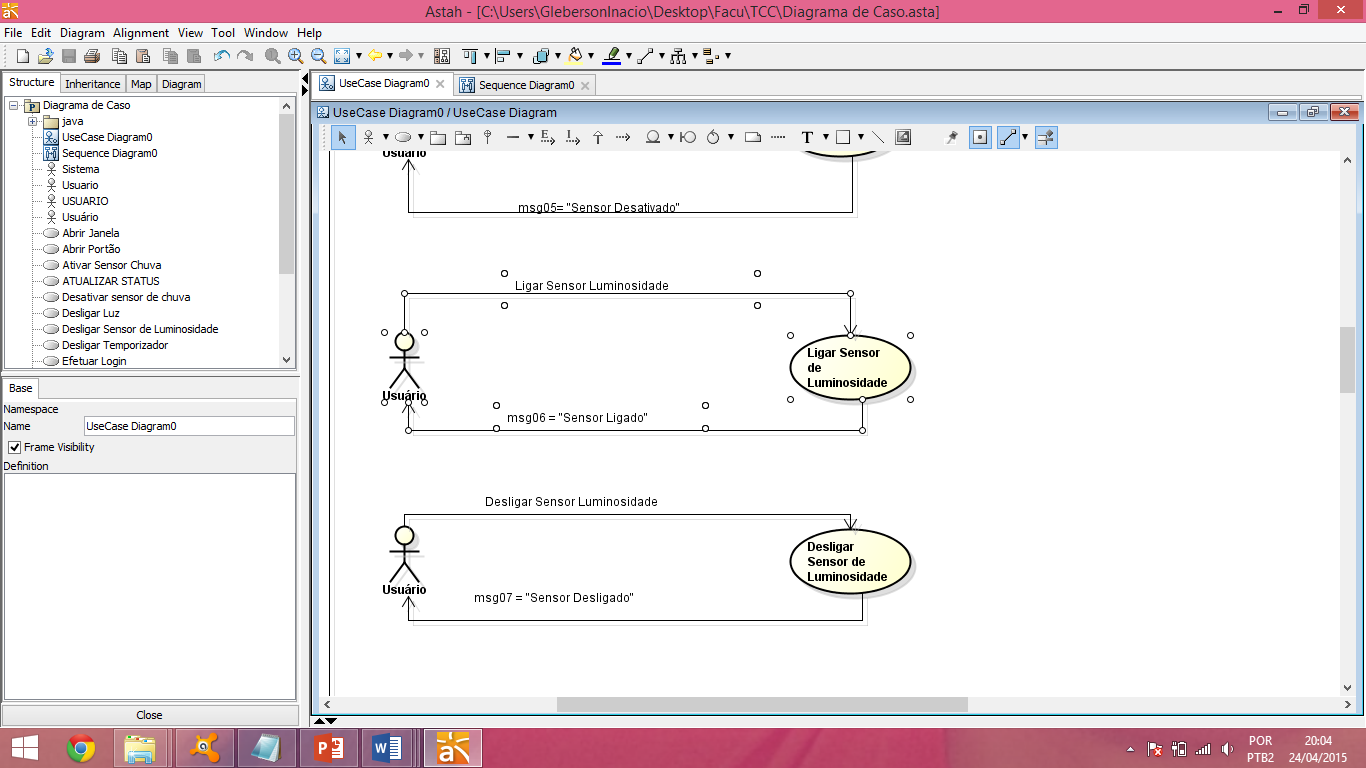


Figura 14 – Diagrama de Caso de Uso Ligar Sensor de Luminosidade

**Cenário principal:**

1. Usuário seleciona a tela de sensores.
2. Usuário seleciona sensor de luminosidade.
3. Usuário liga o sensor.
4. Sistema envia dados para o WebService.
5. WebService valida o status grava no banco e devolve para o sistema / <<include>> Gravar Log.
6. Sistema Emite MSG06.

**Cenário Alternativo:**

Sensor já ativado

5.1. WebService Valida Sensor já ativado e retorna status para o sistema.

5.1.2 Sistema emite MSG 06 “O sensor já está ativado e retorna ao passo 2”.

**Diagrama de Sequência:**

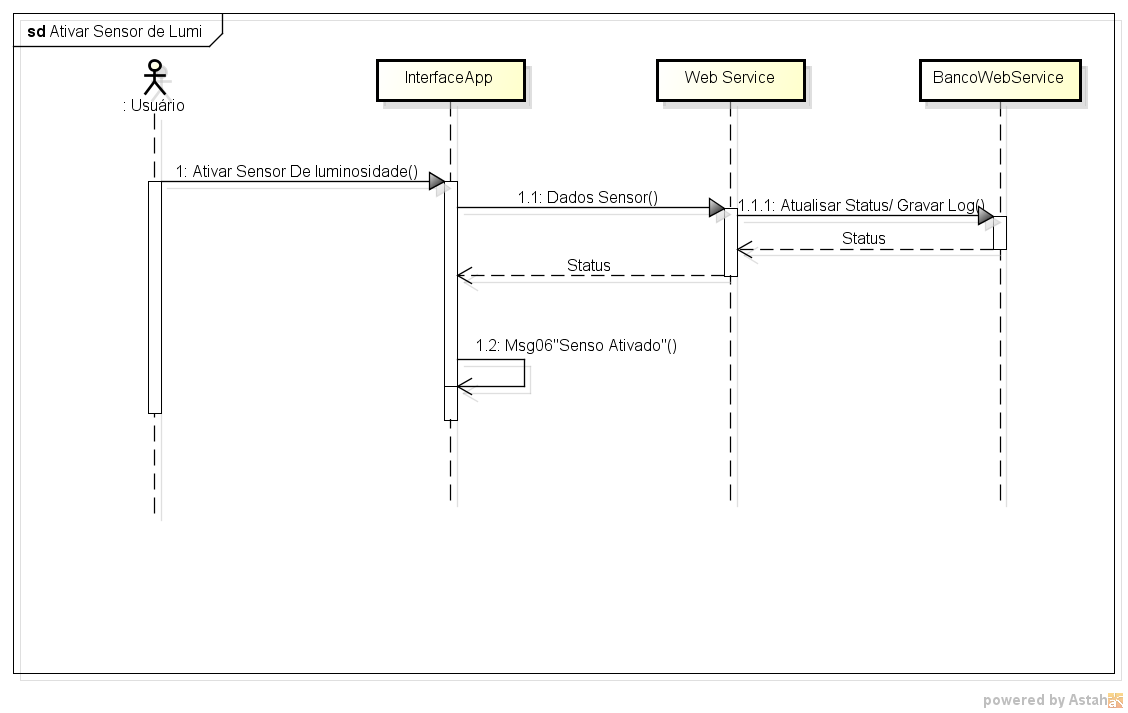


Figura 15 – Diagrama de Sequencia Ligar Sensor de Luminosidade

**Caso de Uso 7** – Desativar Sensor De Luminosidade

**Objetivo:** Desativar o sensor de luminosidade.

**Ator:** Usuário

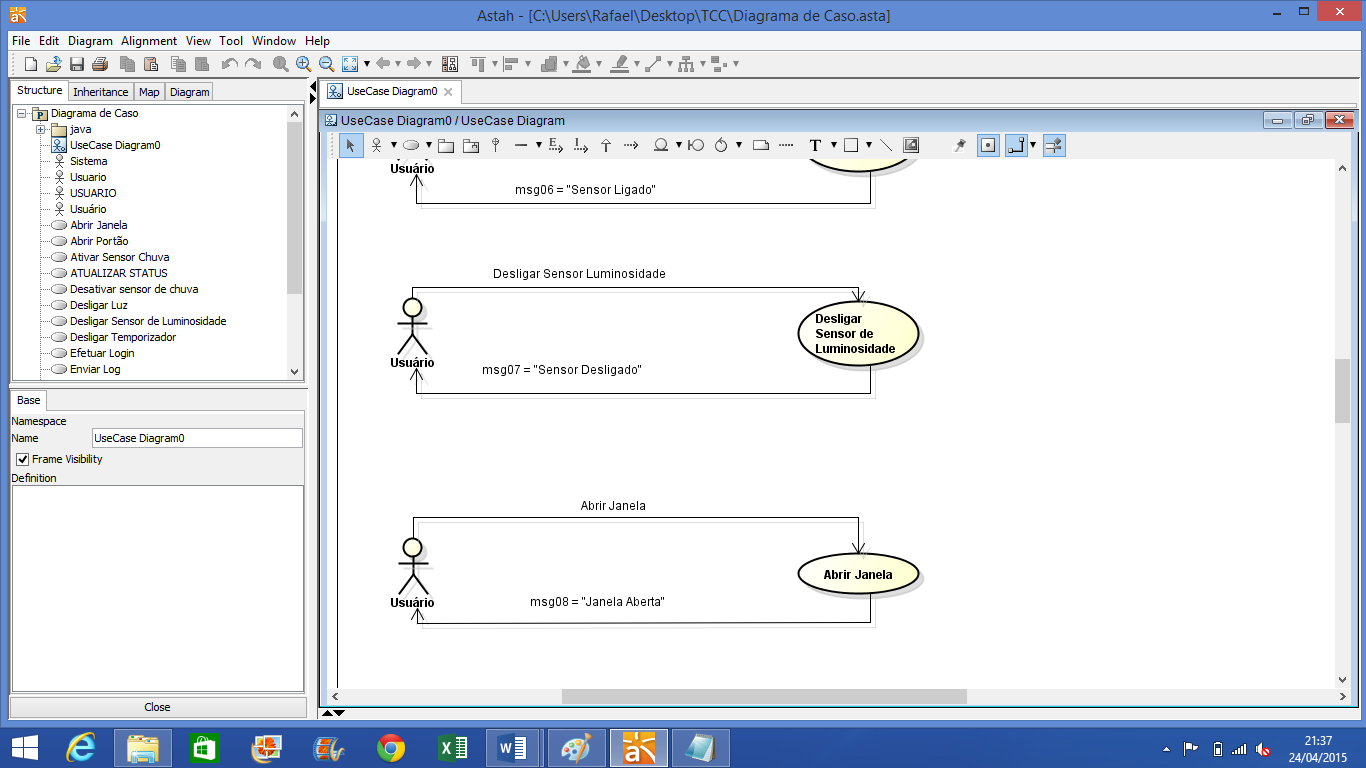


Figura 16 – Diagrama de Caso de Uso Desativar Sensor de Luminosidade

**Cenário Principal**

1. Usuário seleciona a tela de sensores.
2. Usuário seleciona sensor de luminosidade.
3. Usuário Desliga o sensor.
4. Sistema dados para o WebService
5. WebService Valida o status grava no banco e devolve ao sistema. /<<include>> Gravar log.
6. Sistema emite MSG06

**Cenários Alternativos**

Sensor já Desativado

5.1 WebService Valida Sensor Desativado e retorna Status ao sistema.

5.2 Sistema emite MSG06 “O sensor já está Desativado” e retorna ao passo 2.

**Diagrama de Sequência**

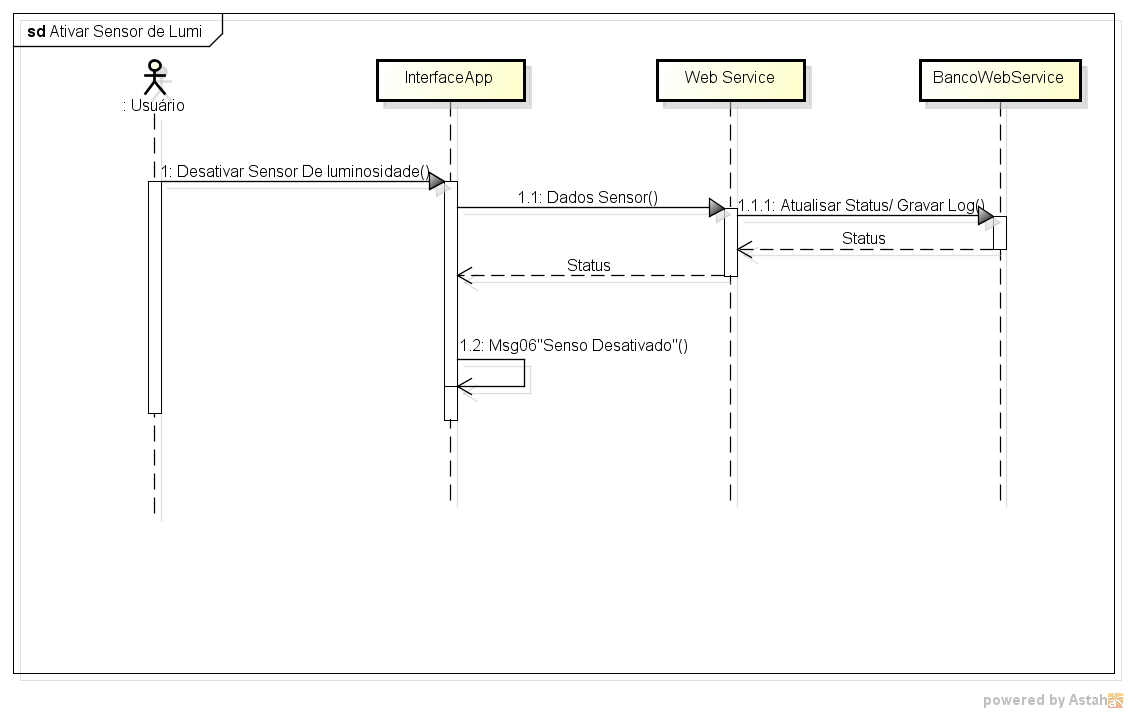
****

Figura 17 – Diagrama de Sequencia Desligar Sensor de Luminosidade

**Caso de uso 8** – Gravar log

**Objetivo:** Gravar toda interação e alteração no sistema por parte do usuário.

**Ator:** Sistema

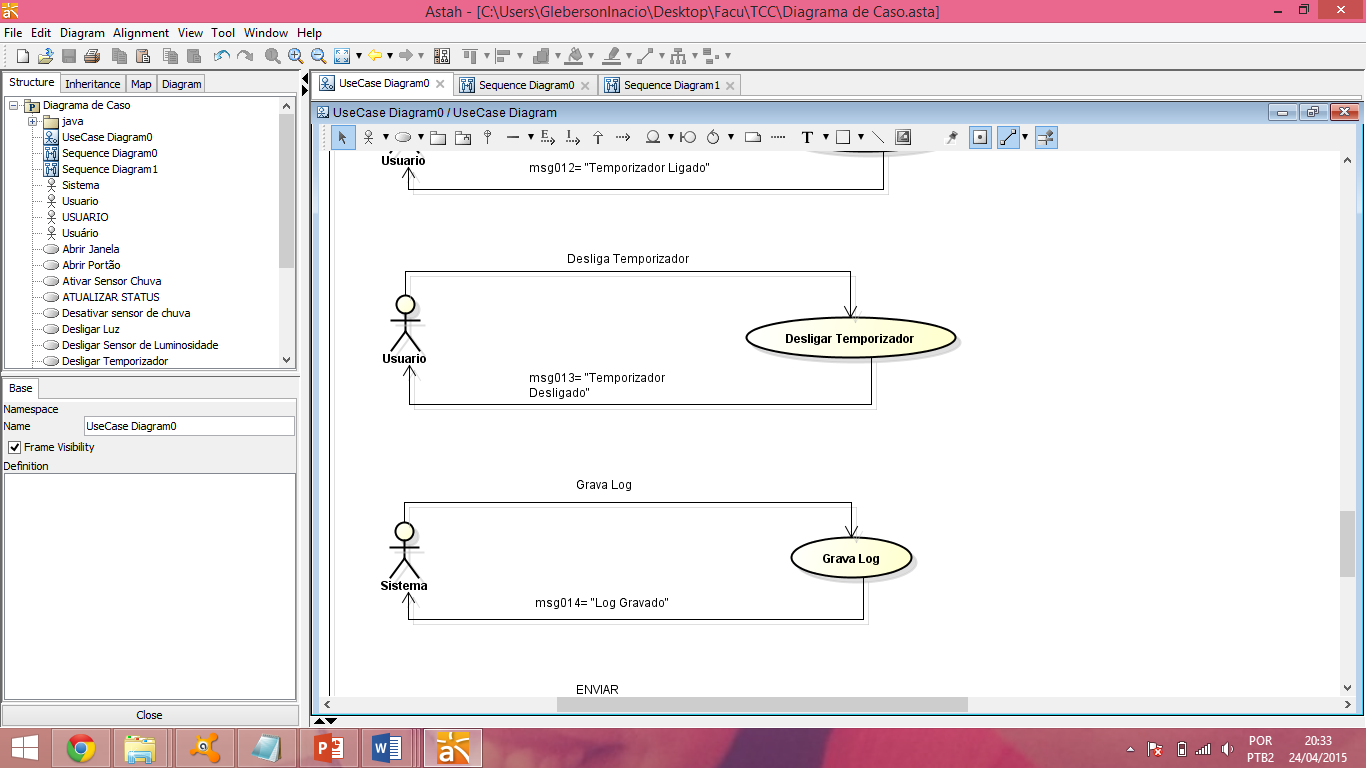


Figura 18 – Diagrama de Caso de Uso Gravar Log

**Cenário principal:**

1. Sistema registra toda a interação que ocorre na casa.
2. Sistema envia o registro para o WebService
3. WebService Grava o Log no banco.

**Diagrama de Sequência:**

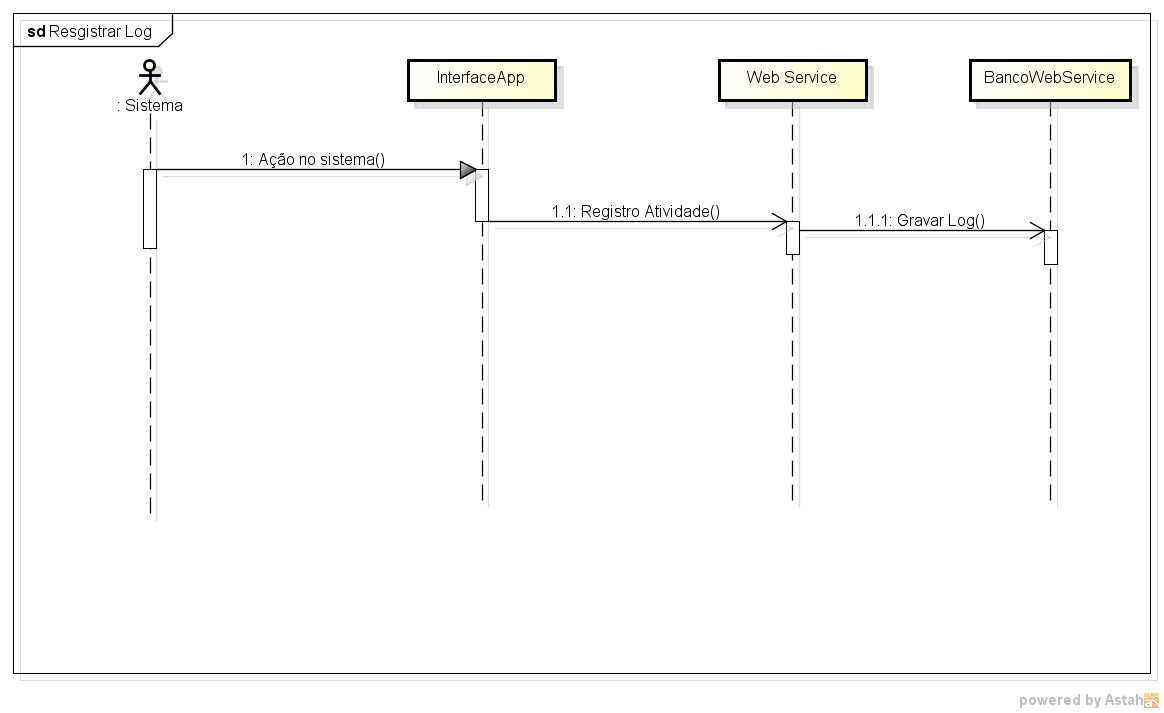


Figura 19 – Diagrama de Sequencia Gravar Log

**Caso de uso 9** – Atualizar Status

**Objetivo:** Atualizar a situação de todos os componentes ligados ao sistema.

**Ator:** Usuário

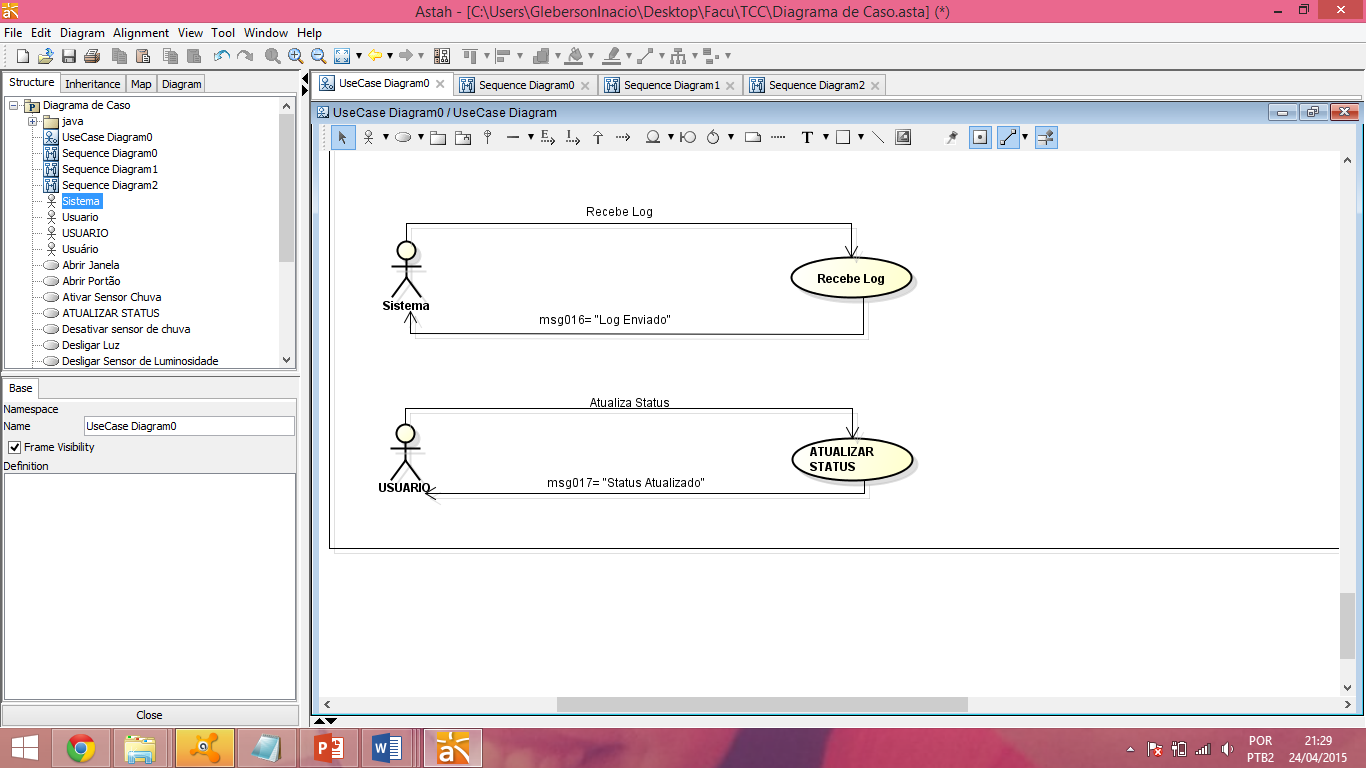


Figura 20 – Diagrama de Caso de Uso Atualizar Status

**Cenário principal:**

1. Usuário seleciona a opção atualizar status na tela inicial.
2. Sistema envia a solicitação ao WebService.
3. WebService envia a solicitação ao banco de dados e a retorna ao sistema.
4. Sistema emite MSG17.

**Diagrama de Sequência:**

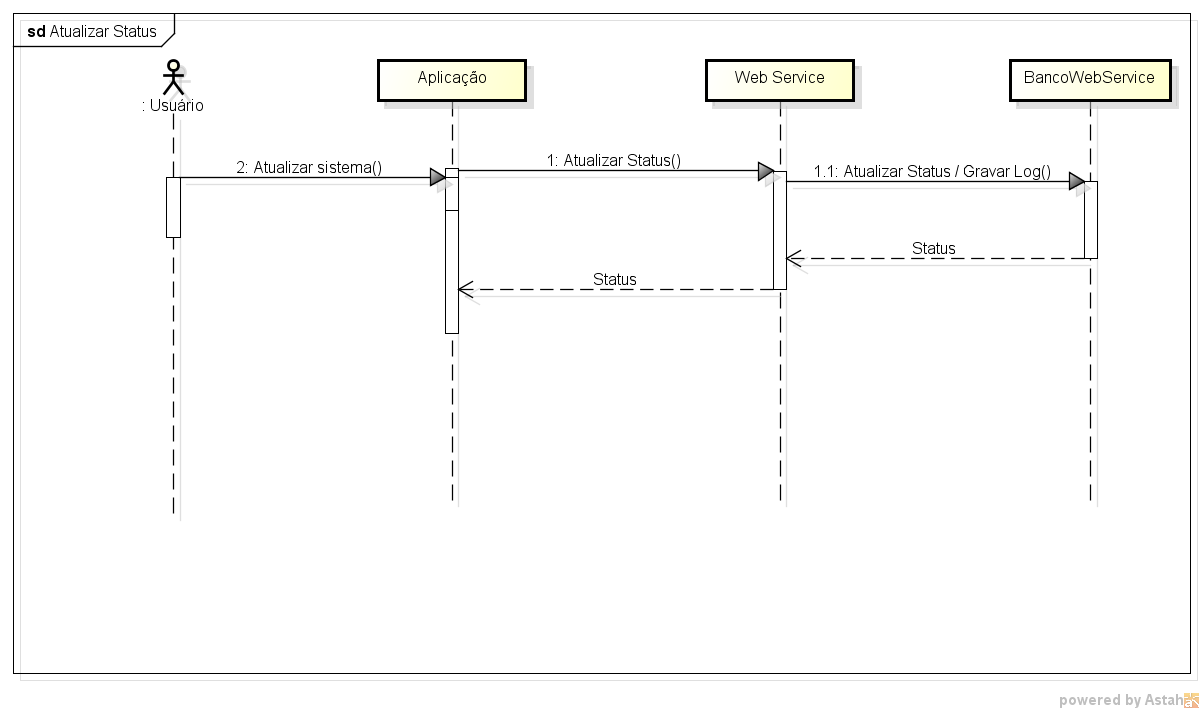


Figura 21 – Diagrama de Sequencia Atualizar Status

3. PROJETO DO SISTEMA

* 1. Ambiente de Desenvolvimento

O sistema foi desenvolvido para a plataforma androide, usando o Sistema Operacional Windows. O gerenciador de banco de dados usado foi o Mysql. A linguagem de desenvolvimento do sistema foi Java para androide, tendo como IDE de desenvolvimento o Eclipse Luna.

* 1. Modelo conceitual de banco de dados

3.2.1 Modelo Conceito de banco de dados (Web Service)

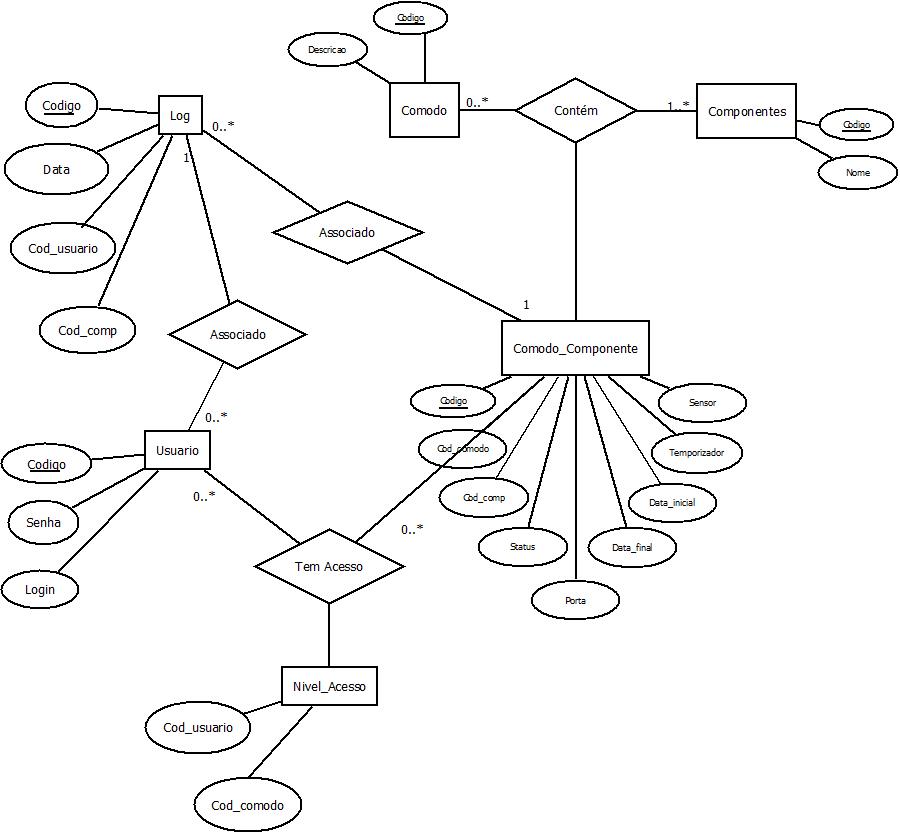


Figura 22 – Modelo Conceitual do Banco de Dados (Web Service)

3.2.2 Modelo Conceito de banco de dados (Android)

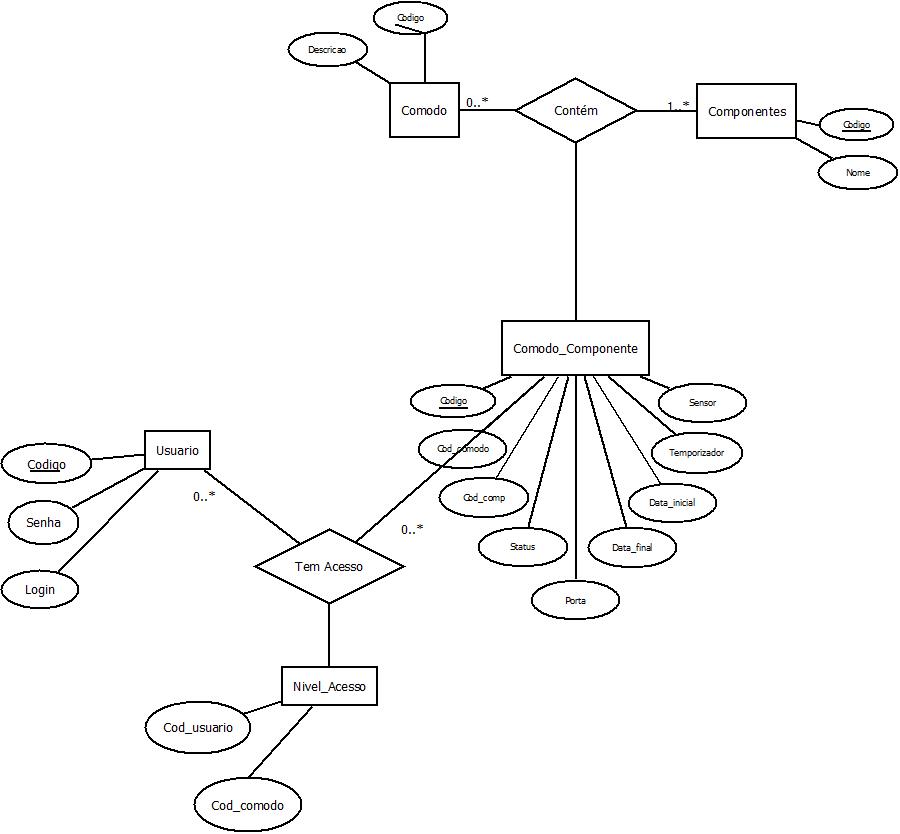


Figura 23 – Modelo Conceitual do Banco de Dados (Web Service)

* 1. Modelo Logico Do Banco De Dados (Classes)

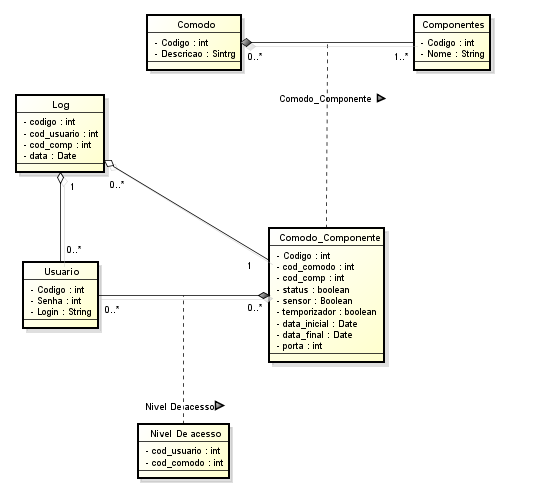


Figura 24 – Modelo Logico do Banco de Dados (Web Service)

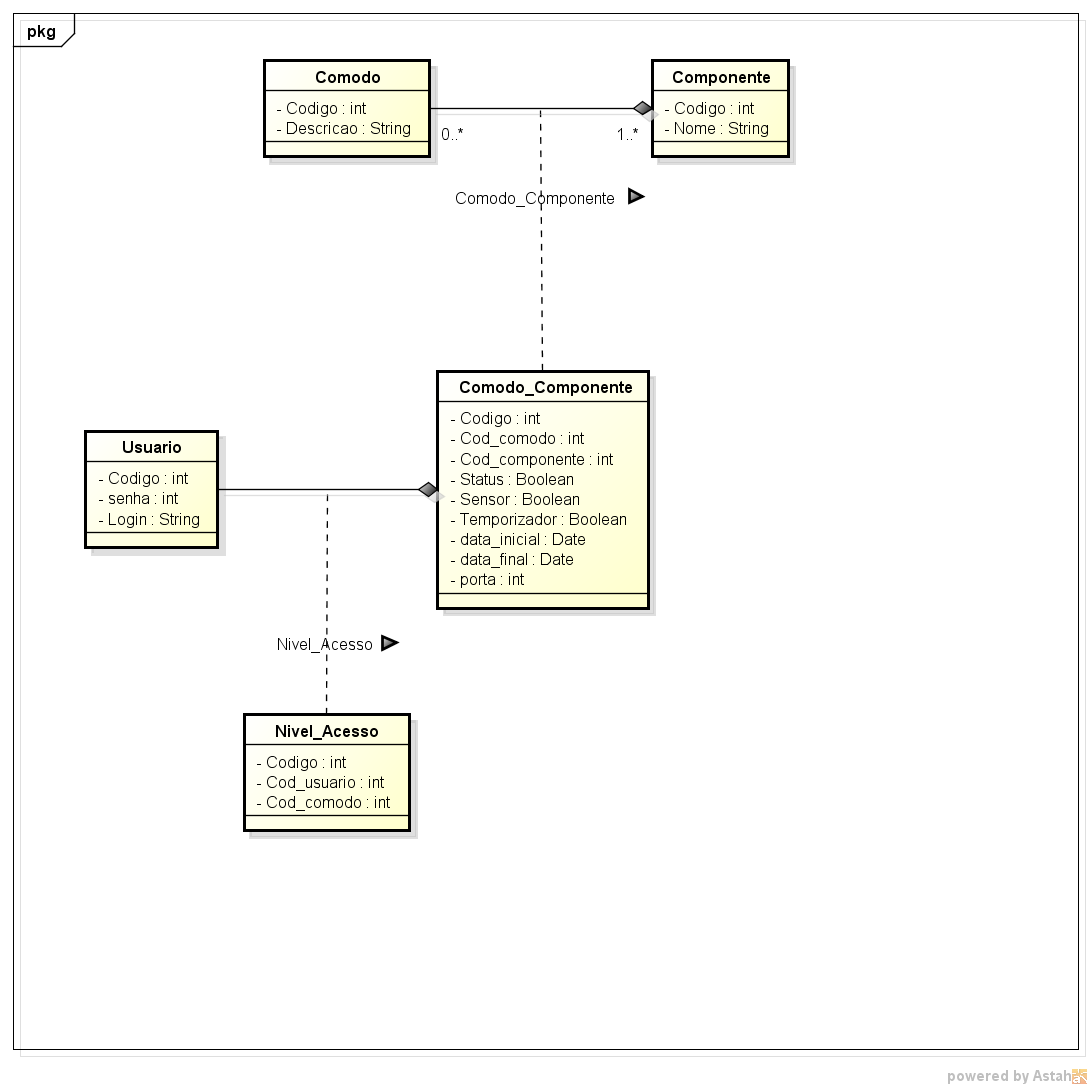


Figura 25 – Modelo Logico do Banco de Dados (Android)

* 1. Modelo Físico de Banco de Dados (Tablas e Scripts)

3.4.1 Script do Web Service

CREATE TABLE `usuario` (

`idusuario` int(11) NOT NULL AUTO\_INCREMENT,

`login` varchar(45) NOT NULL,

`senha` varchar(10) NOT NULL,

PRIMARY KEY (`idusuario`)

)

CREATE TABLE `componente` (

`idcomponente` int(11) NOT NULL AUTO\_INCREMENT,

`descricao` varchar(50) NOT NULL,

PRIMARY KEY (`idcomponente`)

)

CREATE TABLE `comodo` (

`idcomodo` int(11) NOT NULL AUTO\_INCREMENT,

`descricao` varchar(50) NOT NULL,

PRIMARY KEY (`idcomodo`)

)

CREATE TABLE `comodo\_componente` (

`idcomodo\_componente` int(11) NOT NULL AUTO\_INCREMENT,

`idcmodo` int(11) NOT NULL,

`idcomponente` int(11) NOT NULL,

`status` tinyint(4) NOT NULL,

`sensor` tinyint(4) NOT NULL,

`temporizador` tinyint(4) NOT NULL,

`data\_inicial` date NOT NULL,

`data\_final` date NOT NULL,

`porta` int(11) NOT NULL,

PRIMARY KEY (`idcomodo\_componente`),

KEY `idcmodo` (`idcmodo`),

KEY `idcomponente` (`idcomponente`),

CONSTRAINT `comodo\_componente\_ibfk\_1` FOREIGN KEY (`idcmodo`)

REFERENCES `comodo` (`idcomodo`),

CONSTRAINT `comodo\_componente\_ibfk\_2` FOREIGN KEY (`idcomponente`)

REFERENCES `componente` (`idcomponente`)

)

CREATE TABLE `log` (

`idlog` int(11) NOT NULL AUTO\_INCREMENT,

`idcomodo\_componente` int(11) NOT NULL,

`idusuario` int(11) NOT NULL,

`data` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON

UPDATE CURRENT\_TIMESTAMP,

PRIMARY KEY (`idlog`),

KEY `idcomodo\_componente` (`idcomodo\_componente`),

KEY `idusuario` (`idusuario`),

CONSTRAINT `log\_ibfk\_1` FOREIGN KEY (`idcomodo\_componente`)

REFERENCES `comodo\_componente` (`idcomodo\_componente`),

CONSTRAINT `log\_ibfk\_2` FOREIGN KEY (`idusuario`) REFERENCES

`usuario` (`idusuario`)

)

CREATE TABLE `nivel\_acesso` (

`idnivel\_acesso` int(11) NOT NULL AUTO\_INCREMENT,

`idcomodo\_componente` int(11) NOT NULL,

`idusuario` int(11) NOT NULL,

PRIMARY KEY (`idnivel\_acesso`),

KEY `idcomodo\_componente` (`idcomodo\_componente`),

KEY `idusuario` (`idusuario`),

CONSTRAINT `nivel\_acesso\_ibfk\_1` FOREIGN KEY (`idcomodo\_componente`)

REFERENCES `comodo\_componente` (`idcomodo\_componente`),

CONSTRAINT `nivel\_acesso\_ibfk\_2` FOREIGN KEY (`idusuario`)

REFERENCES `usuario` (`idusuario`)

)

**3.4.2 Script do Android**

CREATE TABLE `usuario` (

`idusuario` int(11) NOT NULL AUTO\_INCREMENT,

`login` varchar(45) NOT NULL,

`senha` varchar(10) NOT NULL,

PRIMARY KEY (`idusuario`)

)

CREATE TABLE `componente` (

`idcomponente` int(11) NOT NULL AUTO\_INCREMENT,

`descricao` varchar(50) NOT NULL,

PRIMARY KEY (`idcomponente`)

)

CREATE TABLE `comodo` (

`idcomodo` int(11) NOT NULL AUTO\_INCREMENT,

`descricao` varchar(50) NOT NULL,

PRIMARY KEY (`idcomodo`)

)

CREATE TABLE `comodo\_componente` (

`idcomodo\_componente` int(11) NOT NULL AUTO\_INCREMENT,

`idcmodo` int(11) NOT NULL,

`idcomponente` int(11) NOT NULL,

`status` tinyint(4) NOT NULL,

`sensor` tinyint(4) NOT NULL,

`temporizador` tinyint(4) NOT NULL,

`data\_inicial` date NOT NULL,

`data\_final` date NOT NULL,

`porta` int(11) NOT NULL,

PRIMARY KEY (`idcomodo\_componente`),

KEY `idcmodo` (`idcmodo`),

KEY `idcomponente` (`idcomponente`),

CONSTRAINT `comodo\_componente\_ibfk\_1` FOREIGN KEY (`idcmodo`)

REFERENCES `comodo` (`idcomodo`),

CONSTRAINT `comodo\_componente\_ibfk\_2` FOREIGN KEY (`idcomponente`)

REFERENCES `componente` (`idcomponente`)

)

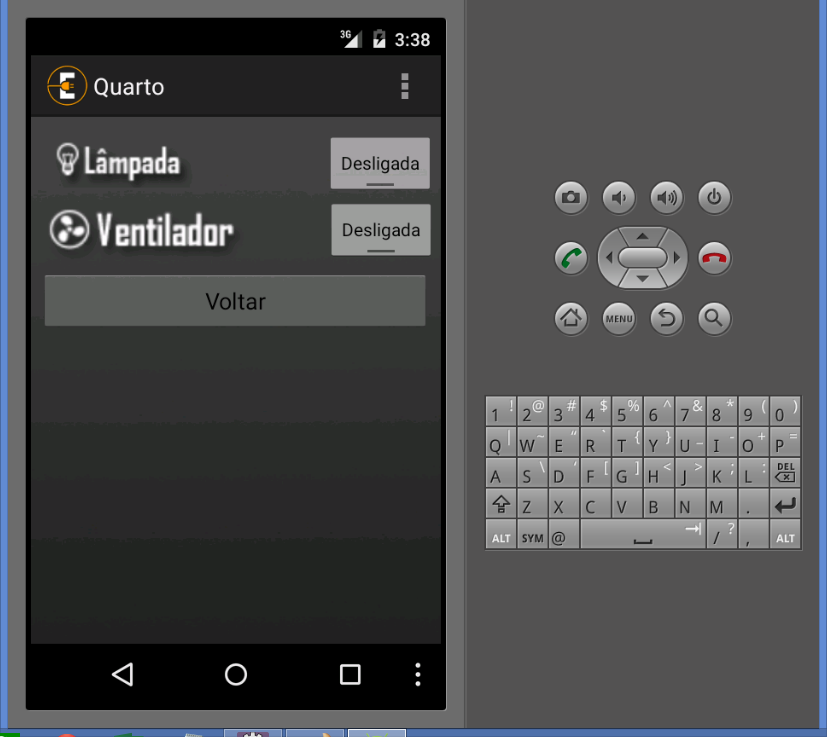
* 1. Modelos de Interface



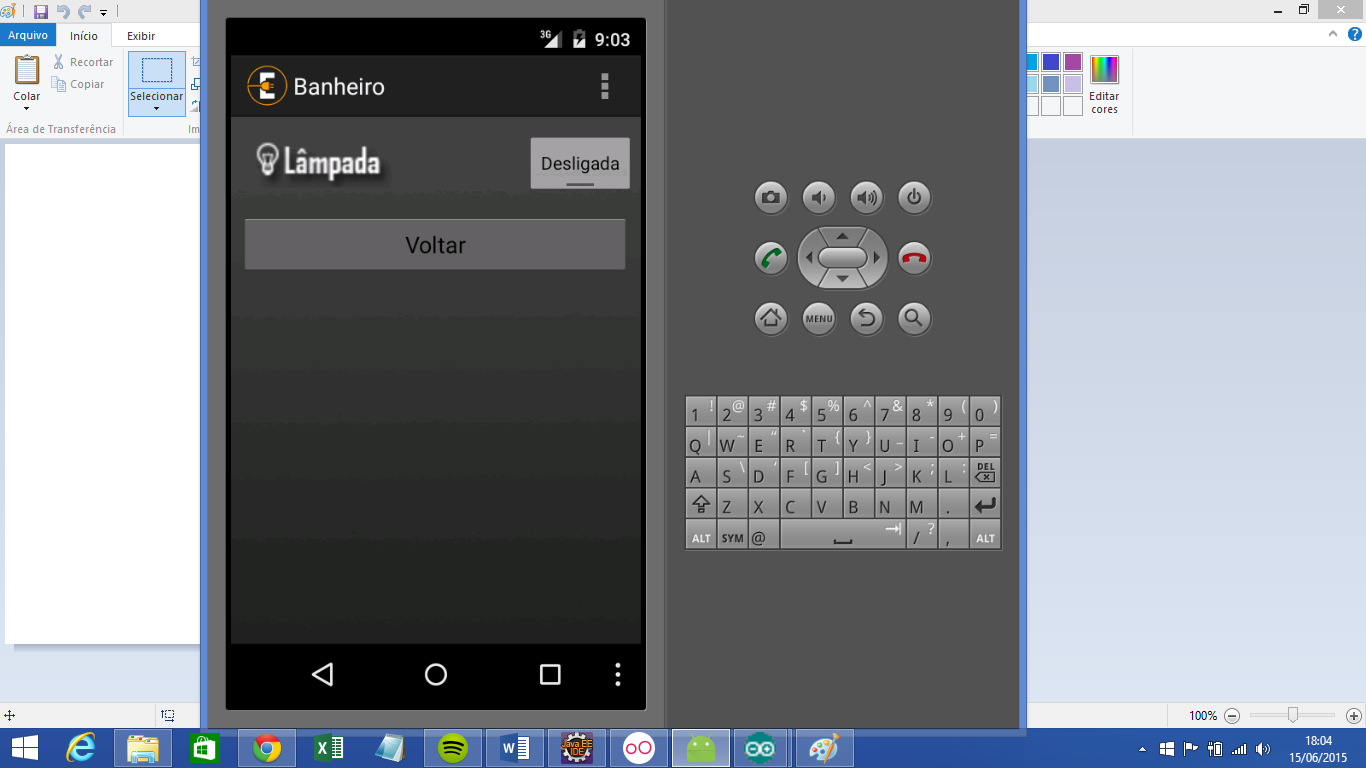
Tela de Login do Sistema.



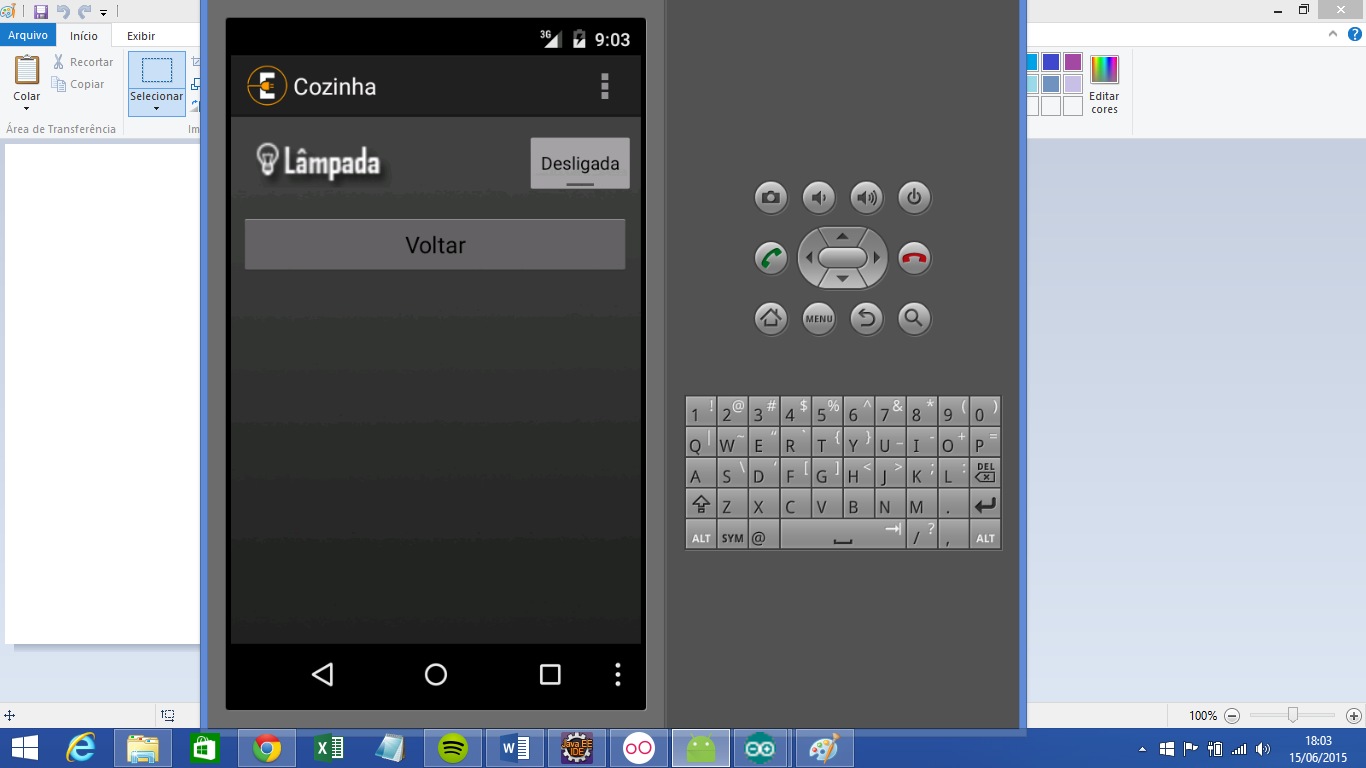
Tela Principal do Sistema



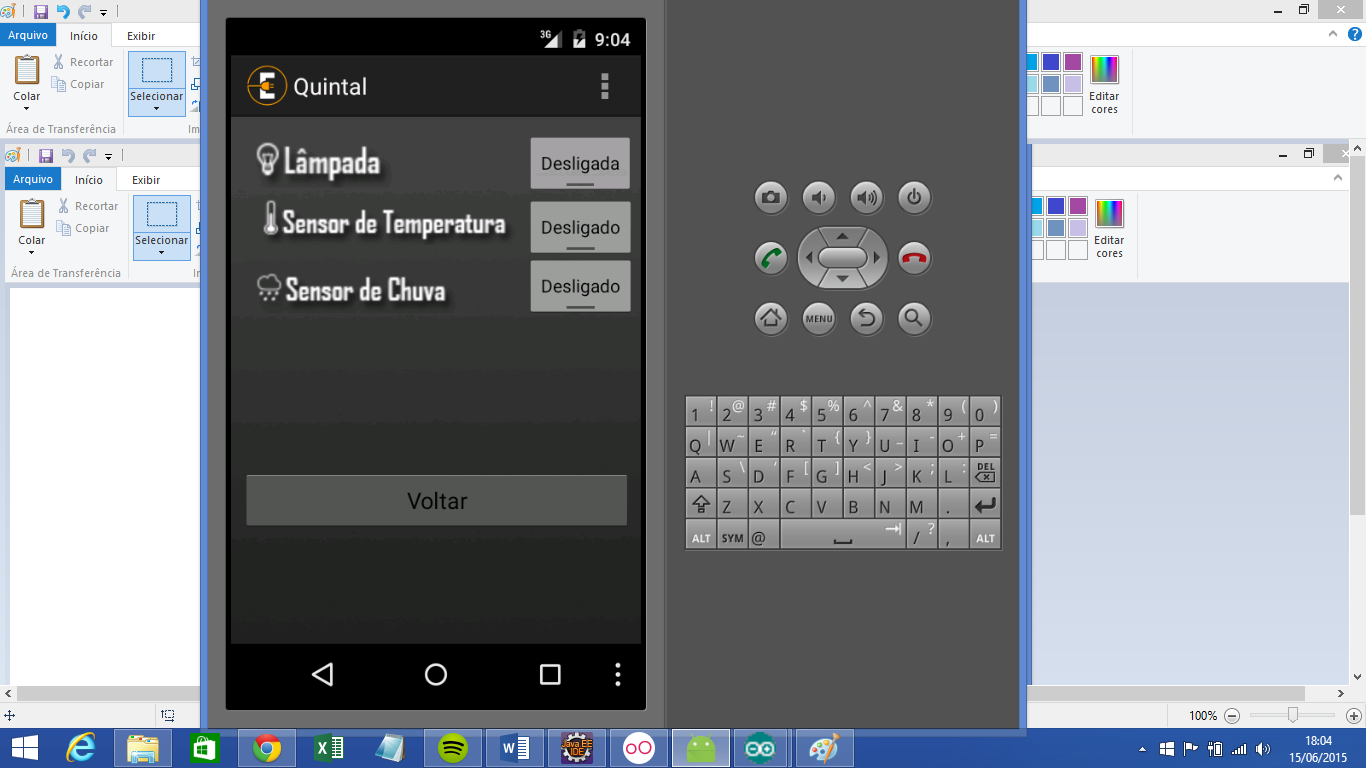
Tela do Quarto



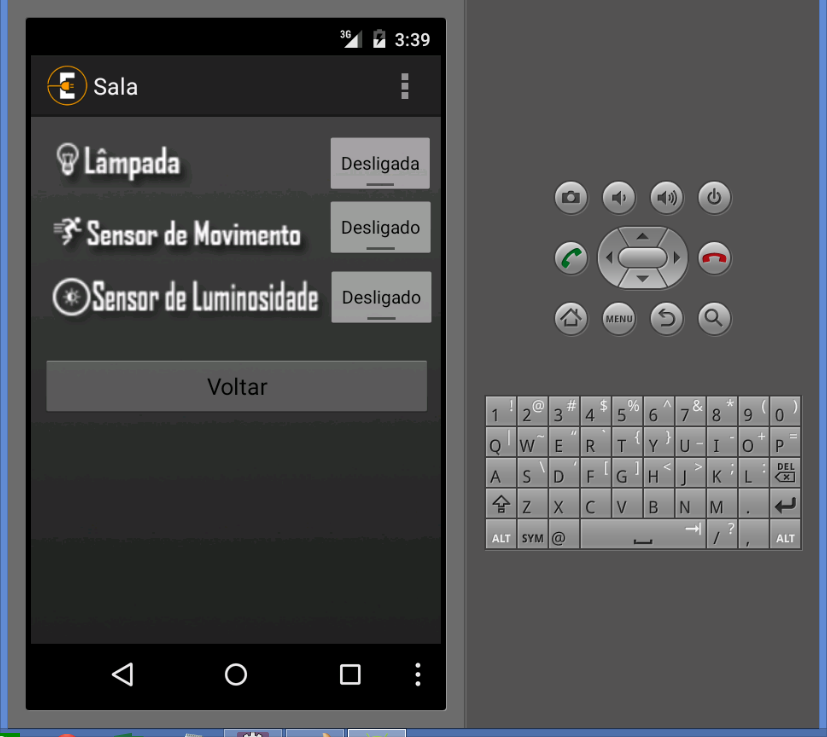
Tela do Banheiro



Tela da cozinha

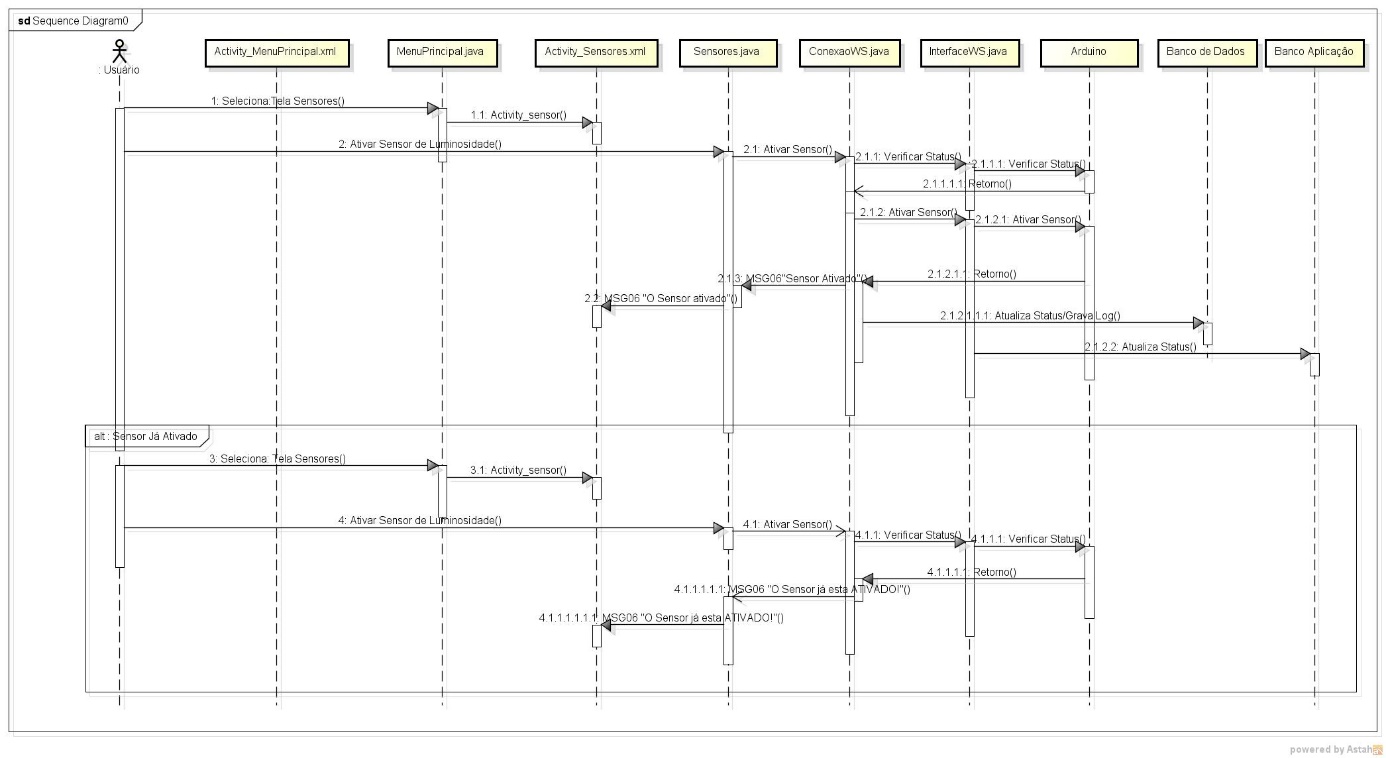


Tela do Quintal

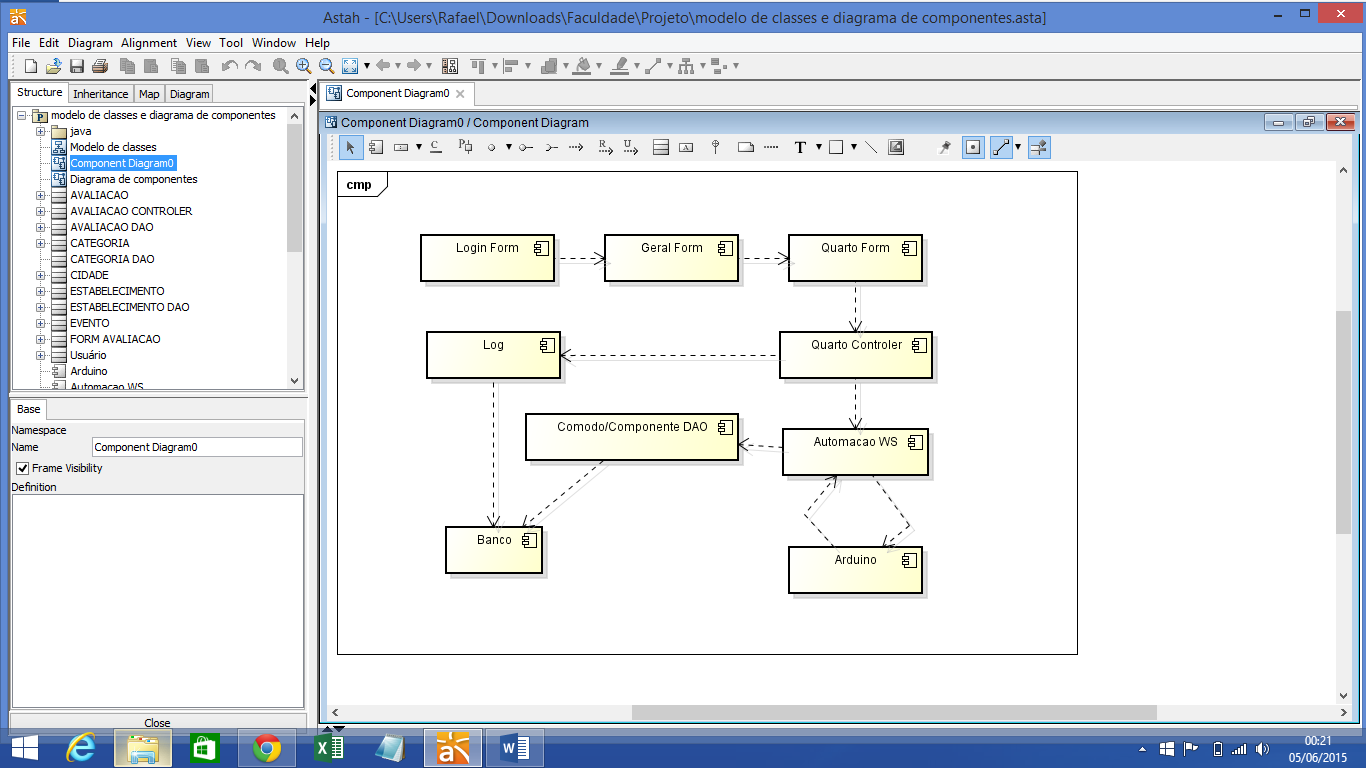


Tela da Sala

* 1. Arquitetura de Classe
  2. Modelo de Sequencia



* 1. Modelo de Componente



# **QUALIDADE E SEGURANÇA DE SOFTWARE**

Conceitos de qualidade são imprecisos e difíceis de serem aceitos por todas as pessoas, no entanto, métricas de qualidade de software surgem desde a década de 70 e vêm se desenvolvendo de forma a ajudar no processo de desenvolvimento de software.

A garantia de controle de qualidade de software está intimamente relacionada a atividades de verificação e validação e estão presentes em todo o ciclo de vida do software. Em algumas organizações não existe distinção entre essas atividades. Entretanto, a garantia de qualidade e os processos de verificação e validação de software devem ser atividades distintas. A garantia de qualidade é uma função gerencial, enquanto que a validação e a verificação são processos técnicos no desenvolvimento de software.

Dentre os modelos de gerenciamento de controle de qualidade de software mais conhecidos estão o Capability Maturity Model (CMM) e o ISO 9000-3, que foram motivados pelas falhas nos processos de gerência e manutenção durante o desenvolvimento de software [CESAR97].

A qualidade de software não é uma idéia tão simples. É mais fácil descrevê-la através de um conjunto de atributos ou fatores requeridos que variam de acordo com as diferentes aplicações e os clientes que as solicitam.

Existem várias formas de se classificar os fatores de qualidade. Uma delas é classificá-los como fatores externos e fatores internos. Fatores externos são aqueles cuja presença ou falta num produto de software pode ser detectada pelos usuários do produto (velocidade, facilidade de uso). Fatores internos são aqueles que são perceptíveis apenas por profissionais de computação (modularidade). Apesar de apenas os fatores externos terem importância no final, a chave para assegurar que eles são satisfeitos são os fatores internos, ou seja, as técnicas internas são um meio para atingir qualidade de software externa. Alguns atributos externos são: corretude, robustez, extensibilidade, reusabilidade, compatibilidade, eficiência, portabilidade, verificabilidade, integridade e facilidade de uso [Meyer88].

Outra maneira de se classificar os atributos de qualidade é dividi-los em atributos funcionais e atributos não funcionais. Os atributos funcionais tipicamente se aplicam a pedaços do software, módulos do sistema como um todo e estão mais relacionados com o que deve ser feito. Já os atributos não funcionais podem se aplicar a qualquer produto do processo de desenvolvimento: especificações, código, manuais, etc., e estão mais relacionados com o quão bem deve ser feito [McDermid94].

O crescimento de computadores conectados à internet vem aumentando e, consequentemente, oferecendo mais possibilidades de ataques. Isto coloca o software e, consequentemente, as organizações e usuários em grande risco. (MCGRAW, 2006)

O acompanhamento dessa evolução da Internet permitiu-nos desenvolver novos recursos para controles de segurança e de privacidade. Mesmo com estes diversos recursos e controles sofisticados na infraestrutura de redes e junto ao usuário final, o software sempre foi e será o principal vetor de ataque.

Atualmente, esse cenário se torna cada vez mais complexo, pois a possibilidade de integração e de acoplamento de novos recursos aos softwares, geralmente chamados de extensões, afeta negativamente a sua segurança, aumentando o risco de acordo com o nível de integrações que a ele oferece. (MCGRAW, 2006)

Sabendo da exposição das empresas e dos usuários que utilizam a internet, este trabalho propõe uma maneira de integrar os conceitos de segurança da informação, principais atividades e melhores práticas de segurança de software junto à metodologia ágil SCRUM para a construção de softwares seguros.

A segurança de informação visa a garantir a confidencialidade, integridade e disponibilidade das informações, além de garantir a conformidade com a legislação vigente e a continuidade dos negócios. De forma mais ampla, pode-se definir como prática de gestão de riscos e de incidentes evitar o comprometimento dos três principais conceitos da segurança, descritos abaixo:

**Confidencialidade** é o conceito que define que toda informação deve ser protegida de forma que o acesso não autorizado não seja permitido.

**Integridade** é o conceito que define que toda a informação deve ser mantida da mesma forma que foi disponibilizada, visando à proteção contra quaisquer alterações.

**Disponibilidade** é o conceito que define que toda informação disponibilizada por um indivíduo ou instituição deve estar acessível no momento em que houver necessidade para qualquer finalidade. Com base nos conceitos apresentados acima a noção de risco de segurança de software tem se tornado de conhecimento comum, mas os programadores, arquitetos e cientistas da computação só agora começaram a estudar sistematicamente como criar um software seguro. (MCGRAW, 2006)

Criação de um sistema seguro requer um processo de segurança integrado no ciclo de desenvolvimento do software. Este processo de segurança de software deve cobrir desde a arquitetura até a implantação do projeto e, principalmente, definir o papel dos profissionais de segurança de software dentro do ciclo de desenvolvimento de software, de modo que o processo seja aplicado de forma eficaz. (SWIDERSKI e SNYDER, 2004)

No sistema Elektra, optamos por realizar uma tela de login, onde o dono da residência ou familiar acessa com seu usuário e senha pré cadastrados para assim ter acesso aos cômodos do sistema.

A imagem abaixo representa o login do sistema de automação residencial via android Elektra:



1. CONCLUSÃO

A automação residencial se apresenta na atualidade como uma revolução nos ambientes domésticos por incorporar esse novo conceito de integração entre os diversos equipamentos e dispositivos de uma casa numa única central de comando. Apesar do ceticismo que ainda existe por parte dos consumidores, pode-se perceber que cada vez mais a sociedade e usuários demandam por soluções de automação em suas residências com vistas à automatização de pequenas tarefas diárias e repetitivas, aumento da segurança e entretenimento.

Com isso, podemos concluir que a automação residencial se configura num desafio do presente, devendo prover ao usuário interfaces amigáveis e descomplicadas, como também disponibilizar a informação e possibilidade de controle da residência a partir de qualquer lugar, através da Internet, de modo a utilizar a eletrônica como “plano de fundo” para colocar em primeiro plano a sociabilidade e bem-estar do usuário.

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

Associação Brasileira de Automação Residencial - AURESIDE. Disponível em: <www.aureside.org.br>. Acesso em: 28/03/2015

OLIVEIRA, A. M.; Automação Residencial. Monografia apresentada ao Departamento de Ciências da Administração e Tecnologia, do Centro Universitário de Araraquara. 2005.

Cassiane de Fátima dos Santos Bueno – Disponível em: <http://www.cin.ufpe.br/~mrsj/Qualidade/Qualidade%20de%20Software.pdf>. Acesso em: 02/06/2015.

Rego, Bruno Motta – Disponível em:<http://pt.slideshare.net/BrunoMottaRego/segurana-no-desenvolvimento-de-sistemas-com-metodologia-gil-scrum?qid=c327ca88-6ab0-4a22-ac0b-38c76b0b5cc8&v=qf1&b=&from\_search=1>. Acessado em: 02/06/2015.

1. **CRONOGRAMA**

Cronograma das atividades feita pelos alunos em cada mês.

|  |  |  |
| --- | --- | --- |
| **Mês** | **Etapa** | **Atividades Realizadas** |
| Março | - Levantamento de Requisitos | - Visão Geral do Sistema  - Problemas Existentes  - Escopo do Sistema  - Requisitos Funcionais  - Requisitos não funcionais |
| Abril | - Análise e Modelagem de Requisitos  - Modelo de Classes do Domínio de Negócio  - Detalhamento dos Casos de Uso e Diagrama de Sequência | - Lista de Casos de Uso  - Modelos de Caso de Uso  - Diagrama de Classe  - Diagrama de Sequência |
| Maio | - Projeto do Sistema | - Ambiente de Desenvolvimento  - Modelo Conceitual de Banco de dados (DER)  - Modelo Lógico de Banco de dados (Classes)  - Modelo Físico de Banco de Dados (Tabelas e Scripts)  - Modelos de Interface  - Arquitetura de Classes |
| Junho | - Projeto do Sistema | - Modelo de Sequência  - Modelo de Componentes |
| Julho | - Conclusão  - Referências Bibliográficas  - Cronograma | - Conclusão  - Referências Bibliográficas  - Cronograma |