1.0

ANACLARABOT

5 de novembro de 2016

Michael Silva

Sumário

PREFÁCIO

O objetivo deste documento é formalizar a arquitetura do software AnaclaraBot, como: as definições dos requisitos, funcionais e não funcionais, as tecnologias utilizadas em cada módulo do projeto e os métodos de desenvolvimento.

Com esse documento, será possível entender todo o funcionamento do software em uma visão macro, no qual serão ilustrados os principais fluxos e transições.

INTRODUÇÃO

O AnaclaraBot é um software de conversação automatica, ou seja, não é necessário que uma pessoa esteja do outro lado para responder as perguntas de um internauta. Será usada como base a tecnologia *Watson* [1] da IBM. A qual utiliza técnicas cognitivas, que ajudam na interpretação da linguagem natural fornecida pelo internauta, assim como na resposta que o sistema dará.

Então o administrador do sistema alimentará o *Watson*, ligando prováveis respostas a palavras específicas. Com isso, o *back-end* do AnaclaraBot irá consumir os serviços do *Watson* passando como parâmetro o texto que o internauta enviar. Muitas vezes, existirá links contidos nos textos, os links são informações mais completas sobre determinado assunto.

DESCRIÇÃO GERAL DO SISTEMA

1. **Problema:** Com o AnaclaraBot em pleno funcionamento, a empresa poderá reduzir os gastos de pessoal drasticamente. Já que, não será necessário uma equipe de telemarketing para se comunicar com os clientes, salvo em situações bastante específicas. O AnaclaraBot responderá e interpretará as interações com o cliente usando linguagem natural. É muito importante que o administrador do sistema alimente o *Watson* corretamente, abrangindo os assuntos mais importantes, cujo os clientes necessitam de respostas ou informações.

2. **Regras de Negócio:** A versão do *front-end* assim como *back-end* será colocada em um servidor de ultravelocidade, o qual utiliza a tecnologia de *cloud* / *elastic*, fornecido pela empresa *Amazon*. Além dos arquivos de código do software, o BD (Banco de Dados) estará também nesse servidor. Toda a responsabilidade de tolerância a falhas e segurança de armazenamento das informações são de responsabilidade da mantenendora do servidor (*Amazon*).

3. **Usuários do Sistema:** O sistema contempla basicamente 2 usuários: O Adminsitrador, o qual alimenta/treina o *Watson* e o Internauta que interage com o *Watson*, já treinado.

4. **Armazenamento:** Todas as conversas entre os internautas e o *Watson* serão persistidas em um banco de dados. Esses dados, posteriormente, poderão ser usados por alguma ferramenta de *Business Intelligence* para que gere alguma informação relevante para a empresa.

5. **Desenvolvimento:** O desenvolvimento do software segue o padrão MVC, tanto o *front-end* com o AngularJs (View: Layout da aplicação; Controller: Chamadas a api's; Model: Modelos dos objetos a serem enviados nas api's.), quanto o *back-end* com o NodeJs (View: Especificação das api's; Controller: Lógica de armazenamento no banco; Model: Modelo MongoDB para salvar o objeto).

REQUISITOS

1. **Funcionais** O software AnaclaraBot será incluido no site de uma empresa, com isso, quando algum internauta acessar esse site, poderá iniciar uma conversa. A cada frase que esse internauta enviar, ela responderá conforme a identificação das palavras chaves contidas no texto (a acertude das respostas é de total responsabilidade do *Watson*, juntamente com o treinamento que o administrador do sistema forneceu). A cada interação, as motivadas pelo internauta ou as respondidas pela AnaclaraBot serão armazenadas em um Banco de Dados. A descrição está na Figura 1.

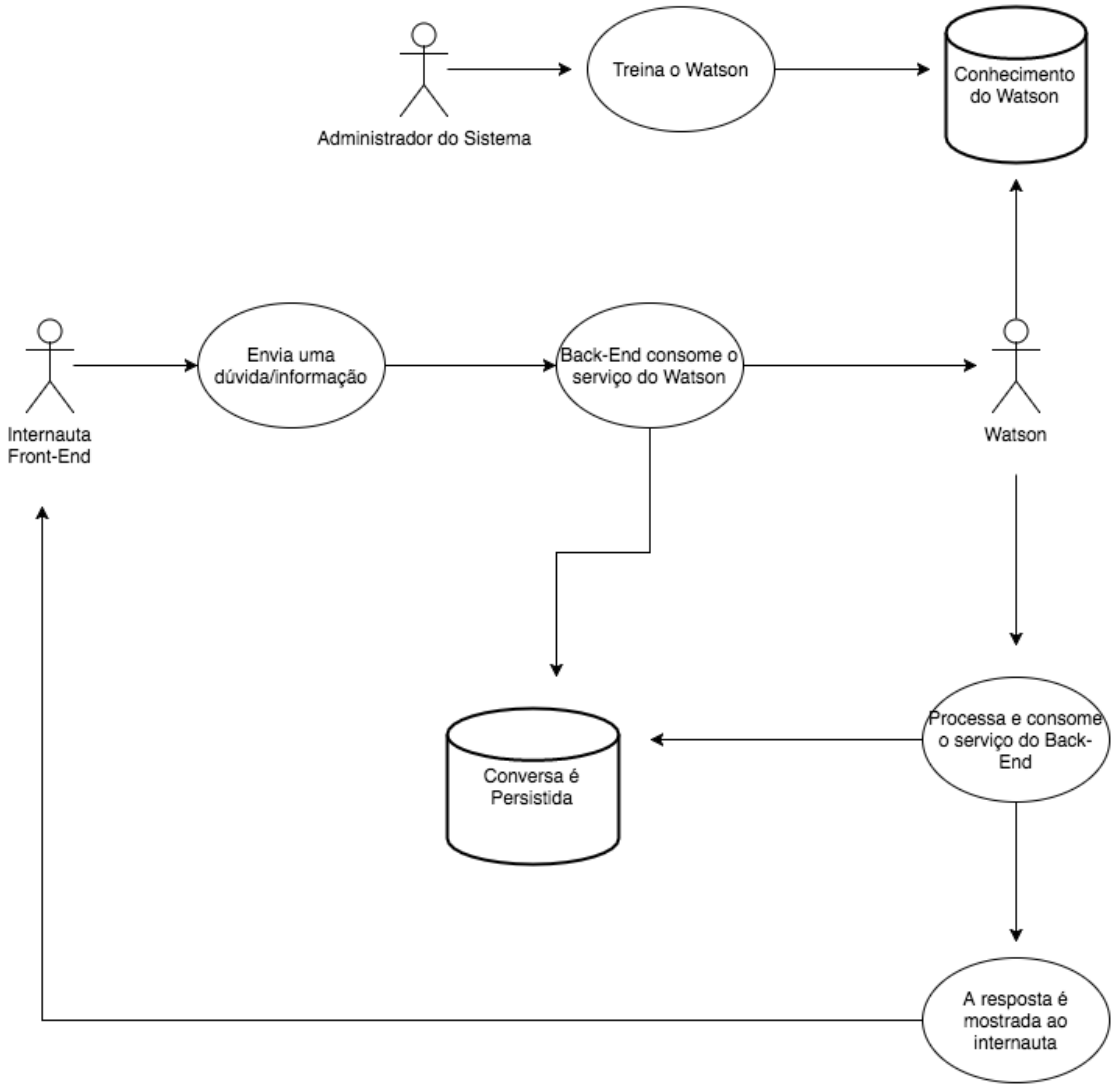

Figura 1: Descrição de funcionament

2. **Não Funcionais**

   (a) O *back-end* estará disponível para as requisições dos internautas, assim como para armazenar as interações, desde que o servidor fornecido por terceiros *Amazon* esteja em pleno funcionamento.

   (b) Terá uma senha, para que os seus serviços sejam consumidos, essa senha será armazenada em variável de ambiente do sistema operacional.

   (c) Somente os administradores do sistema terão as senhas de acesso ao servidor.

   (d) O software será desenvolvido seguindo os padrões de projetos descrita em [2].

   (e) O software será desenvolvido para rodar na seguinte configração mínima: 1 CPU, Linux Ubuntu, 2GB de Memória Ram e Acesso a rede internet. Instância T2 Small da *Amazon* [3].

   (f) O sistema deverá ser acessado completamente via browser HTTP/HTML e via dispositivos Mobile.

CRONOGRAMA DE DESENVOLVIMENTO

Tabela 1: Cronograma de Desenvolvimento da AnaclaraBot

1	Pagamento Inicial: R$ 366,00.	21/10/2016	
2	Arquitetura do Projeto	24/10/2016	
3	Arquitetura do Banco de Dados	26/10/2016	
4	Criação das API's	31/10/2016	Pagamento da parcela número 2.
5	Persistência do JSON	02/11/2016	
6	Design do Front-End em imagem	05/11/2016	Pagamento da parcela número 3.
7	Codificação do Front-End	09/11/2016	Pagamento da parcela número 4.
8	Criar interface para trocar foto da atendente	14/11/2016	
9	Conectar o Front-End no Back-End	14/11/2016	Pagamento da parcela número 5.

TECNOLOGIAS A SEREM UTILIZADAS

1. **Back-End:** Será utilizado o NodeJs [4].
2. **Front-End:** Será utilizado HTML5, CSS3, jQuery, AngularJS, JS e Express [5].
3. **Documentação das Api's:** Todas as API's produzidas pelo *back-end* estarão documentadas com o Swagger [6].
4. **Layout:** O layout será desenhado com o Adobe Photoshop [7].
5. **Banco de Dados:** O banco de dados será desenvolvido com o MongoDB [8].

ENCAPSULAMENTO

O Software tem duas partes integrantes e plugáveis, o *back-end* e o *front-end*. Ambas as partes podem funcionar separadamente, exercendo seus papéis, ou seja, pode ser conectado outro *front-end* ao *back-end* existente, e vice versa. Desde que siga a arquitetura planejada.

O *back-end* pode servir várias instâncias de *fron-end*, estando nos mesmos servidores ou não. Isso tem a idéia de containers, é altamente recomendado instalá-lo em um servidor com estrutura *cloud*, o qual seus recursos de hardware e conexão crescem de acordo com a demanda.

É válido ressaltar que podemos separar cada parte do Software em containers, sendo o BD um container, o *back-end* outro, por último, o *fron-end*. Podem ficar em servidores separados (recomendado) ou no mesmo servidor (não é recomendado). Conforme a Figura 2.

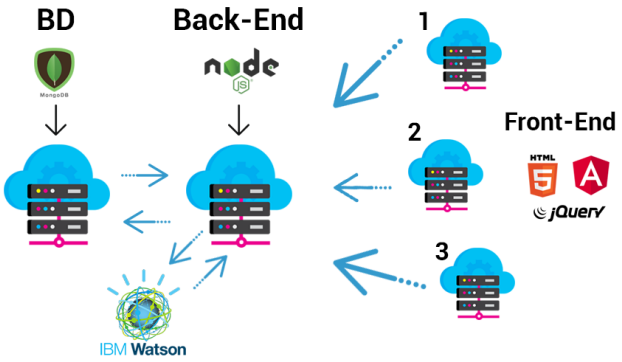

Figura 2: Arquitetura do Software

ARQUITETURA DO BANCO DE DADOS

O Banco de Dados do AnaclaraBot tem 3 coleções, são elas:

1. **User:** Contém os dados dos usuários que se cadastraram, e as referências para as conversas (*Conversations*), as quais ele participou.

2. **Conversation:** Contém os meta-dados das conversas, referências para as interações (*Interactions*) e um ou vários *quizzes* que o usuário responde ao final do chat.

3. **Interaction:** Contém a pergunta do usuário, resposta do bot, os *intents* e as *entities*. Tanto a pergunta do usuário quanto a resposta do bot podem ser em formato de texto, áudio ou imagem.

O modelo do Banco de Dados está descrito na Figura 3.

Figura 3: Modelo do Banco de Dados

Os exemplos da estrutura no formato MongoDB estão descrito nas Listings 1, 2 e 3

```
{
  "userId"       : 1,
  "firstName"      : "Luiz",
  "lastName"       : "Medeiros",
  "email"        : "luiz.medeiros@cognitivesolutions.com.br",
  "cep"        : "72000-000",
  "conversations"    : ["12329-asdasd21-asdaad", "kopsa-09877-
    asd-asdad"]
}
```

Listing 1: Exemplo da estrutura User de dados já no formato MongoDB

```
{
  "conversationId"     : "862c31dc-f06a-461d-860c-e36290c0b6e6
    ",
  "date"           : "2016-10-06",
  "timeInitial"        : "08:00:27",
  "timeEnd"        : "08:13:54",
  "automaticallyClosed" : true,
  "user"           : 8372163732467,
  "interactions"       : [231231,937283,1231212],
  "quiz"           : [
              {
                "type"       : "multiple",
                "question"     : "Quais seus dias da semana
                  preferidos?",
                "choices"    : [
                  {
                    "choice"     : "Segunda",
                    "userSelected"   : false
                  },
                  {
                    "choice"     : "Terca",
                    "userSelected"   : true
```

```
                    },
                    {
                      "choice"    : "Quarta",
                      "userSelected"  : true
                    },
                    {
                      "choice"    : "Quinta",
                      "userSelected"  : true
                    }
                  ]
                },
                {
                  "type": "binary",
                  "question"    : "Voce gostou do atendimento?"
                     ,
                  "choices"   : [
                    {
                      "choice"    : "Sim",
                      "userSelected"  : false
                    },
                    {
                      "choice"    : "Nao",
                      "userSelected"  : false
                    },
                    {
                      "choice"    : "Nao sei",
                      "userSelected"  : true
                    }
                  ]
                }
                ]
}
```

Listing 2: Exemplo da estrutura Conversation de dados já no formato MongoDB

```
{
  "interactionId"     : 2131,
  "userTextInteraction"   : "vacina",
  "userImageInteraction"  : null,
  "userSoundInteraction"  : null,
  "botInteraction"      : "Sempre mantenha [...]",
  "botImageInteraction" : null,
  "botSoundInteraction" : null,
  "date"          : "2016-11-02T16:29:09.000Z",
  "intents"       : [
                {
                  "intent"    : "Aeroporto",
                  "confidence"  : "0.16709978332713935"
                },
                {
                  "intent"    : "vacinacao",
                  "confidence"  : "1"
                }
                ],
  "entities"        : [
                {
                  "entity"    : "Aeroporto",
                  "location"    : [0,4],
                  "value"     : "mala"
                },
                {
                  "entity"    : "vacinacao",
                  "location"    : [0,6],
                  "value"     : "cartao de vacinacao"
                }
                ]
}
```

Listing 3: Exemplo da estrutura Interaction de dados já no formato MongoDB

SCRIPT DE CRIAÇÃO DO BANCO DE DADOS

```
var UserSchema   = new mongoose.Schema({
    firstName: String,
    lastName: String,
    email: String,
    cep: String,
    conversations: [String]
});
```

Listing 4: User Collection

```
var EntitySchema   = new mongoose.Schema({
    entity: String,
    location: [Number],
    value: String
});


var IntentSchema   = new mongoose.Schema({
    intent: String,
    confidence: Number
});


var InteractionSchema   = new mongoose.Schema({
    interactionType: String,
    userTextInteraction: String,
    userImageInteraction: Buffer,
    userSoundInteraction: Buffer,
    botTextInteraction: String,
    botImageInteraction: Buffer,
    botSoundInteraction: Buffer,
    date: String,
    intents: [IntentSchema],
    entities: [EntitySchema]
});
```

Listing 5: Interaction Collection

```
var ChoiceSchema    = new mongoose.Schema({
    choice: String,
    userSelected: Boolean
});


var QuizSchema    = new mongoose.Schema({
    type: String,
    question: String,
    choices: [ChoiceSchema]
});

var ConversationSchema   = new mongoose.Schema({
    conversation_id: String,
    date: String,
    timeInitial: String,
    timeEnd: String,
    automaticallyClosed: Boolean,
    interactions: [{
        type: mongoose.Schema.Types.ObjectId,
        ref: 'Interaction'
    }],
    user: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'User'
    },
    quiz: [QuizSchema]
});
```

Listing 6: Conversation Collection

```
var ClerkSchema   = new mongoose.Schema({
    image: { data: Buffer, contentType: String }
});
```

Listing 7: Clerk Collection

ANACLARABOT EM IMAGENS

FUTURAS IMPLEMENTAÇÕES

1. **Gerenciamento de Aplicações e Segurança:** Será necessário, para a consistência de várias instancias do software, um gerenciamento de aplicações. Com esse gerenciamento, o administrador poderá criar novas instâncias, designando para as mesmas, uma chave e uma senha. Então, cada módulo front-end, terá uma chave de acesso, quando for acessar o back-end, essa chave e senha serão validadas, caso estejam corretas, será gerado um token, e com esse token o front-end poderá acessar as apis.

   Hoje, por motivos de segurança, o acesso a as apis está limitado a só uma origem: http://cognitivesolutionsbr.com. Isso quer dizer que, não serão aceitas requisições de outras fontes.

REFERÊNCIAS

1. Watson. Disponível em: <http://www.ibm.com/watson/> [Accessed Outubro de 2016]

2. Design Patterns. Disponível em: <https://github.com/FredKSchott/the-node-way> [Accessed Outubro de 2016]

3. Amazon. Disponível em: <https://aws.amazon.com/pt/ec2/instance-types/> [Accessed Outubro de 2016]

4. NodeJs. Disponível em: <https://nodejs.org/en/> [Accessed Outubro de 2016]

5. ExpressJs. Disponível em: <http://expressjs.com/pt-br/> [Accessed Outubro de 2016]

6. Swagger. Disponível em: <http://swagger.io/> [Accessed Outubro de 2016]

7. Photoshop. Disponível em: <https://www.adobe.com/br/products/photoshop.html> [Accessed Outubro de 2016]

8. MongoDB. Disponível em: <https://www.mongodb.com/> [Accessed Outubro de 2016]