UFMG – ICEx - DCC
Disciplina: Internet de Coisas

Introdução ao desenvolvimento para RFID

Prof. José Marcos Nogueira
Apresentador: Ewerton Monteiro Salvador

UFMG

UFMG - ICEx
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Plataforma de desenvolvimento

- Alien ALR-9900 e Alien ALR-9650 Development Kits
- Ler e programar tags EPC Class 1 Generation 2, e emitir relatórios de eventos para um sistema em um computador hospedeiro
- O computador hospedeiro pode se ligar ao leitor através da interface RS-232 ou remotamente através de uma interface de rede Ethernet.

Plataforma de desenvolvimento

- Documentação disponibilizada:
  - **RFID Primer** – introdução à tecnologia RFID e um glossário
  - **Reader Interface Guide** – apresentação das interfaces de comunicação do ALR-9900
  - **Quick Installation Guide** – guia rápido para instalação e execução do leitor ALR-9900
  - **Quick Reference** – um guia de referência rápida que sumariza o conjunto de comandos do Alien Reader Protocol
  - **Demo Software Guide** – detalhes da instalação e operação do software de demonstração Alien RFID Gateway
  - **Quick Upgrade Guide** – explica brevemente como usar o software de demonstração para atualizar o ALR-9900

Suporte ao desenvolvimento Java

- A biblioteca Java é fornecida juntamente com o kit de desenvolvimento, através do arquivo "*AlienRFID.jar*" (está no diretório JARs no CD)

- Biblioteca está dividida em 5 grupos funcionais
  - **reader**: classes para comunicação com o leitor
  - **tags**: classes para tratamento de tags e dados EPC
  - **discovery**: classes para se descobrir a localização dos leitores conectados via RS-232 ou Ethernet
  - **notify**: classes para se escutar a notificações "push" e fluxos de dados dos leitores pela LAN
  - **util**: classes para efetuar operações de bits, conversão entre hex/ASCII/strings binárias, parsers XML, gerencialmento de portas seriais, e para determinar a versão da API

Suporte ao desenvolvimento Java

- A comunicação serial não é uma biblioteca que naturalmente acompanha o Java, ela precisa ser adquirida separadamente
- Fornecida gratuitamente pela Sun (pacote javax.comm), com suporte para Solaris e Linux
- Esse pacote deixou de oferecer suporte ao Windows nas versões mais recentes

Reader Classes

- Tipicamente, um objeto *reader* é obtido de *DiscoveryItem*
  - *AlienClass1Reader reader = discoveryItem.getReader();*
- *AbstractReader* é a classe mais genérica desse tipo, e fornece as funções mais básicas. As subclasses são:
  - ***AlienClass1Reader***
  - ***AlienClassOEMReader***
  - ***AlienClassBPTReader***
- Se a localização do leitor é conhecida, pode-se instanciar um reader diretamente
  - AlienClass1Reader reader = new AlienClass1Reader();
  - reader.setConnection("COM1");

Reader Classes

- Instanciação direta via rede:
  - *AlienClass1Reader.setSerialConnection("COM1");*
  - *AlienClass1Reader.setNetworkConnection("1.2.3.4", 23);*
  - *AlienClass1Reader.setNetworkConnection("1.2.3.4:23");*
- Instanciação via construtor da classe:
  - *AlienClass1Reader reader = new AlienClass1Reader("COM1");*
  - *AlienClass1Reader reader = new AlienClass1Reader("1.2.3.4:23");*
- Operações de abertura e fechamento:
  - *reader.open();*
  - *reader.isOpen();*
  - *reader.close();*

Reader Classes

- Modo mais básico de comunicação com o leitor é via comando *doReaderCommand()*
- Equivalente a passar comandos via terminal, com o resultado do comando retornando ao programa como o "retorno" do método
- *String readerName = reader.doReaderCommand("get ReaderName");*
  - ReaderName = Alien RFID Reader
- API oferece métodos equivalentes a diversos comandos

Tag Classes

- Após as leituras serem realizadas, os resultados podem ser retornados tanto como um simples objeto *Tag* como um array de objetos *Tag*
  - *Tag[] tagList = reader.getTagList();*

- Exemplos das propriedades existentes num objeto Tag
  - Tag ID – string contendo o código EPC
  - Discover Time – período em que a tag foi vista pela primeira vez
  - Last Seen Time – período em que a tag foi vista pela última vez
  - Count – número de vezes que a tag foi lida
  - Antenna – número da antena que viu a tag pela última vez
  - Protocol – o protocolo “aéreo” usado para leitura da tag

Tag Classes

- **TagUtil class**: métodos para fazer o parsing e decodificar dados de tags (por exemplo, tags representadas em XML)
- **TagTable class**: mantém uma *HashTable* de tags, com métodos para adicionar e remover tags, e também formas de notificar o sistema sobre modificações nessa lista
- **TagTableListener interface**: comunicar mudanças na *TagTable* para outros objetos

Discovery Classes

- Framework fornece as classes *SerialDiscoveryListenerService* e *NetworkDiscoveryListenerService* para detecção de leitores
- Interface *DiscoveryListener* precisa ser implementada para o recebimento dos eventos de descoberta

  - public interface DiscoveryListener {

  public void readerAdded (DiscoveryItem discoveryItem);

  public void readerRenewed(DiscoveryItem discoveryItem);

  public void readerRemoved(DiscoveryItem discoveryItem);

  }

- A classe *DiscoveryItem* fornece meios para que um software identifique e contate os dispositivos descobertos
  - public AlienClass1Reader getReader() throws Exception
- Retorno do método acima é um objeto AlienClass1Reader

Discovery Classes

- Para a descoberta de um leitor em uma porta serial, é necessário se listar as portas disponíveis no computador e interrogá-las, uma a uma, para identificar os possíveis leitores
- Essa tarefa é feita pela classe *SerialDiscoveryListenerService*
- Já no caso da conexão via rede, é necessário se saber que todo leitor da Alien é configurado para emitir mensagens *heartbeat* na sub-rede local, via UDP (mensagem contém um XML com informações do leitor)
- A classe *NetworkDiscoveryListenerService* escuta essas mensagens para identificação do leitor conectado à interface de rede

Notify Classes

- As classes de notificação permitem que o leitor trabalhe de forma autônoma, enviando notificações para o sistema apenas quando uma etiqueta for lida
- Mecanismo também permite *streamed data* (TagStream e IOStream), proporcionando uma comunicação de baixa-latência entre o leitor e o sistema
- A classe chave desse mecanismo é a *MessageListenerService*
- Esse serviço escuta uma porta específica, ao mesmo tempo em que o leitor deve ser configurado para enviar as notificações para essa porta

Notify Classes

- Passos para configuração do serviço de notificação:
  1. Instruir o leitor para enviar as notificações para uma estação – reader.setNotifyAddress("listener.alien.com:3988")
  2. Instruir o leitor sobre as condições nas quais ele deve enviar as notificações (por exemplo, enviar tag list a cada 30 segundos) – reader.setNotifyTime(30);
  3. Configurar as mensagens de notificação para o formato XML – reader.setNotifyFormat(reader.XML_FORMAT);
  4. Finalmente, solicitar ao leitor que se realize leituras no modo autônomo –
     reader.autoModeReset();

     reader.setAutoMode(reader.ON);

- Para que um objeto receba notificações do *MessageListenerService*, ele deve implementar a interface *MessageListener*

Notify Classes

- Um objeto do tipo *Message* encapsula uma coleção de metadados sobre a própria mensagem de notificação, e um array de objetos Tag
- Exemplos de propriedades desses objetos são:
  - ReaderName
  - ReaderType
  - IPAddress
  - MACAddress
  - CommandPort
  - Time
  - Reason
  - StartTriggerLines
  - StopTriggerLines
  - TagList

Introdução ao desenvolvimento para RFID

Perguntas?