TALLINN UNIVERSITY OF TECHNOLOGY

Embedded service oriented microcontroller architecture

Extensible client-server communication architecture for small devices

Master thesis

Denis Konstantinov
IASM111615

Outline

- SOA background
- Embedded server application
- Client Java RPC stub library
- Android Application
- Demo

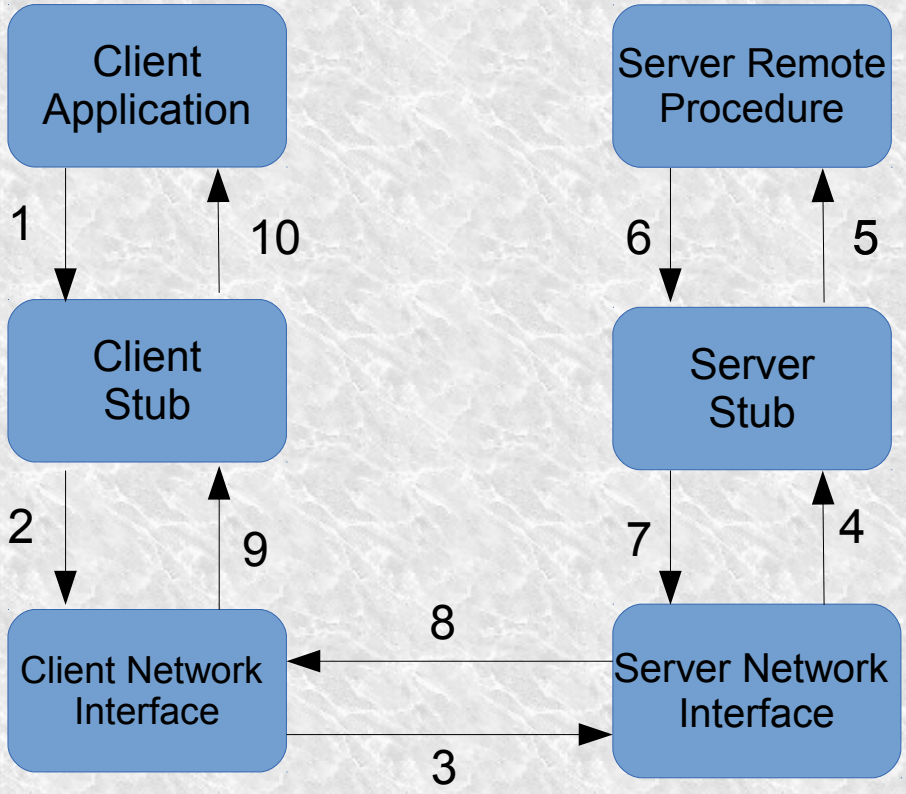
Remote services and Service-Oriented Architecture (SOA)

One of the methods of Machine-to-Machine(M2M) communication

- Unassociated and loosely coupled units of functionality
- Well known communication interface via contracts
- Standard data exchange formats (XML, JSON) and protocols
- Reuse of software modules
- Abstraction and hidden implementation logic
- Multiple hardware platforms
  and software programming languages

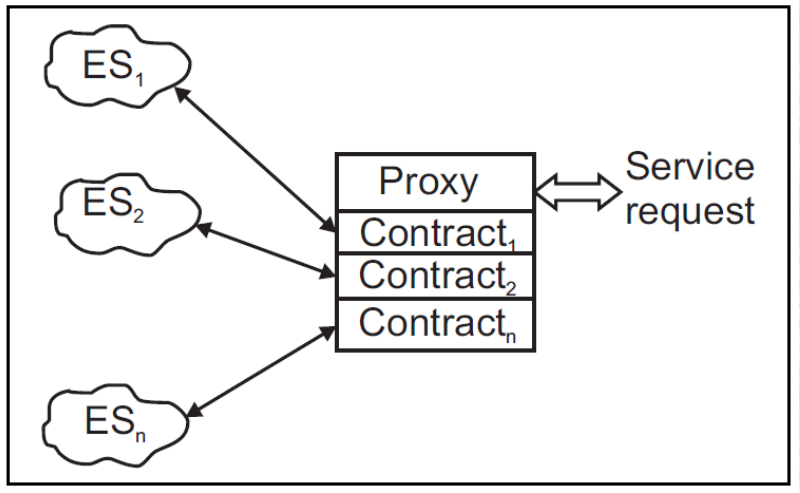
SOA technologies

- Web Services and SOAP
- REST
- **RPC**
- DCOM and WCF
  from Microsoft
- CORBA
- Java RMI
- Jini
- Apache Thrift
- ...

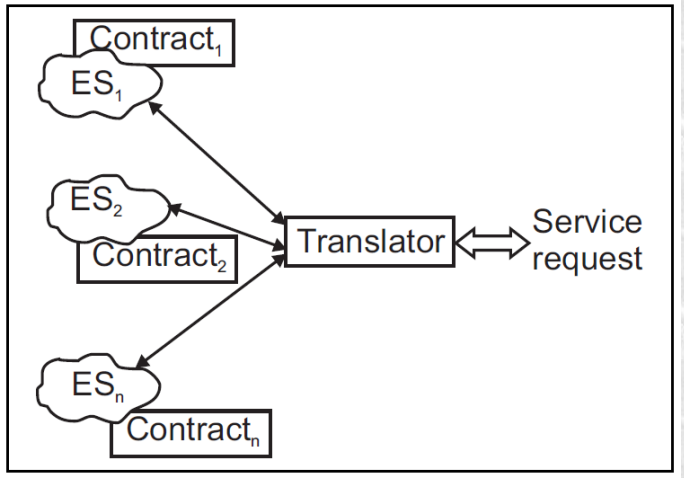


The idea of Remote Procedure Calls

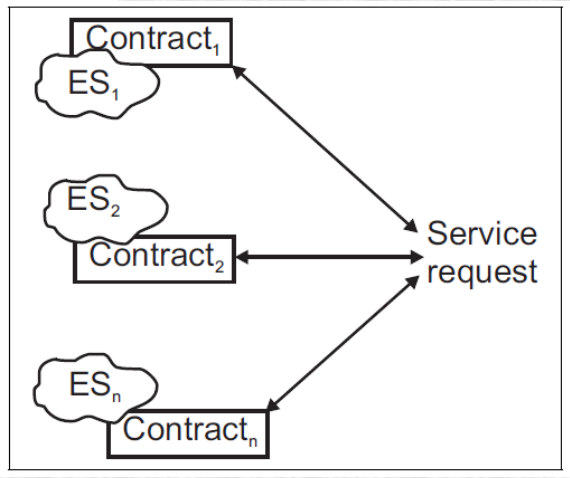
Embedded service-oriented architecture



Proxy architecture



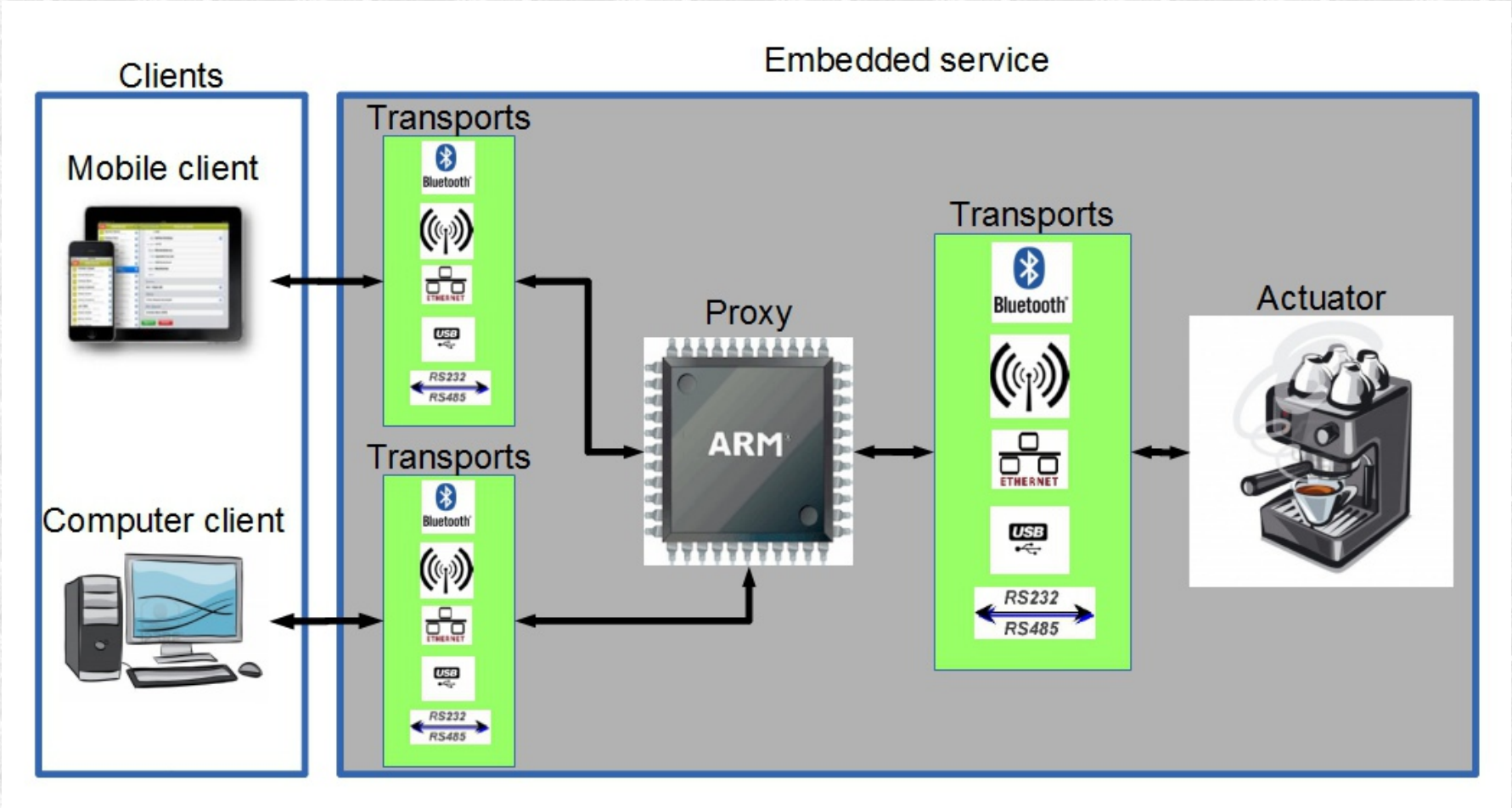
Translator architecture



Full architecture

N. Milanovic, J. Richling, and M. Malek, “Lightweight services for embedded systems,” in Software Technologies for Future Embedded and Ubiquitous Systems, 2004. Proceedings. Second IEEE Workshop on, pp. 40–44, 2004.

Embedded service oriented architecture



Proxy server device

- STM32F1 family ARM Cortex-M3 microcontroller from STMicroelectronics
- 64KB SRAM and 512KB Flash memory
- 5 USART interfaces
- TI LMX9838 Bluetooth-to-serial module
- FTDI USB-to-Serial module
- FreeRTOS
- JSON processing library and JSON-RPC communication

Messages

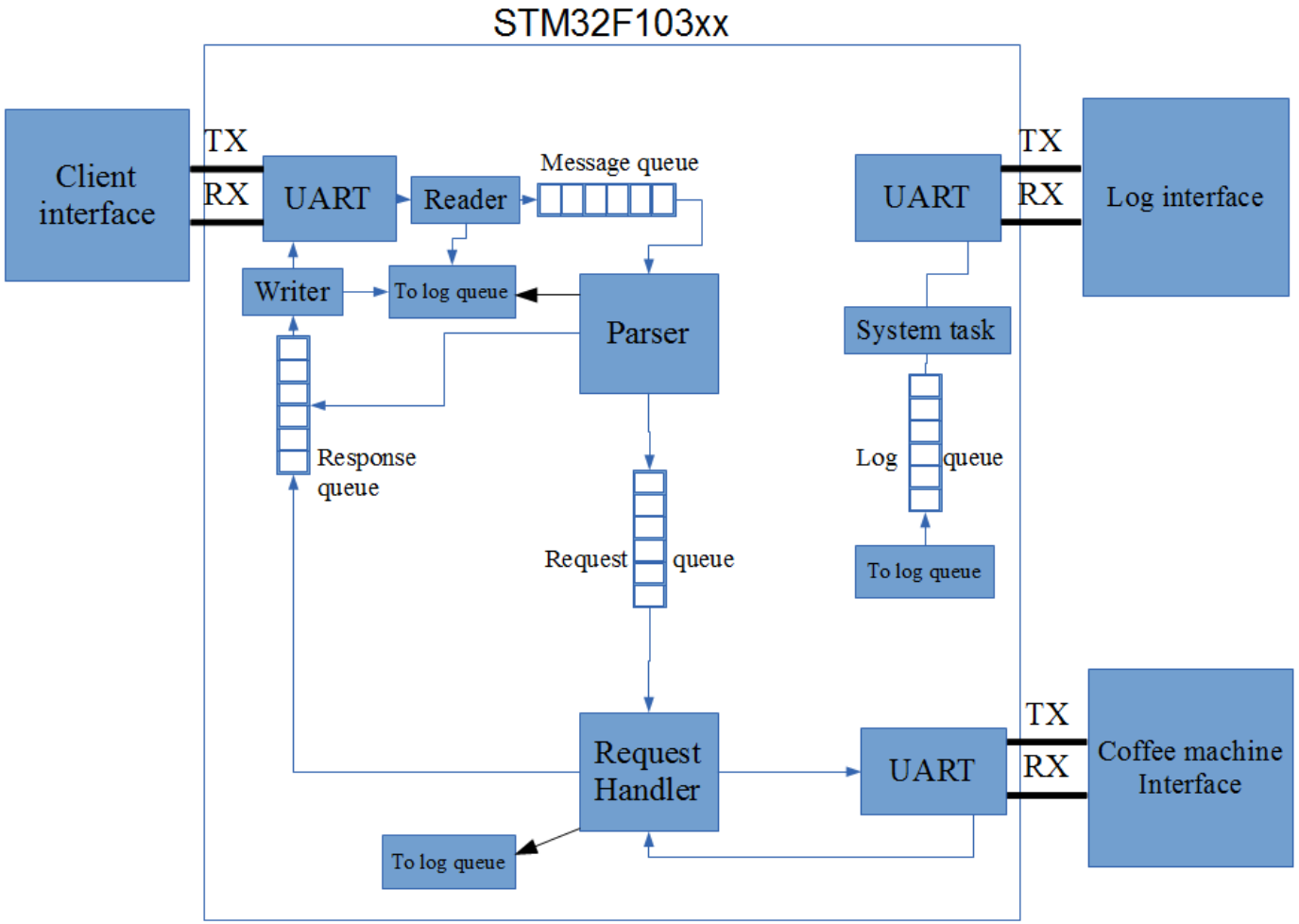
- Netstrings: <LENGTH>:<DATA>,
- Simple JSON-RPC protocol

Legend:
--> data sent to Server
<-- data sent to Client

RPC call with positional parameters:
--> 69:{"jsonrpc": "2.0", "method": "subtract", "params": [42, 23], "id": 1},
<-- 41:{"jsonrpc": "2.0", "result": 19, "id": 1},
--> 69:{"jsonrpc": "2.0", "method": "subtract", "params": [23, 42], "id": 2},
<-- 42:{"jsonrpc": "2.0", "result": -19, "id": 2},

RPC call with named parameters:
--> 94:{"jsonrpc": "2.0", "method": "subtract", "params": {"subtrahend": 23, "minuend": 42}, "id": 3},
<-- 41:{"jsonrpc": "2.0", "result": 19, "id": 3},
--> 94:{"jsonrpc": "2.0", "method": "subtract", "params": {"minuend": 42, "subtrahend": 23}, "id": 4},
<-- 41:{"jsonrpc": "2.0", "result": 19, "id": 4},

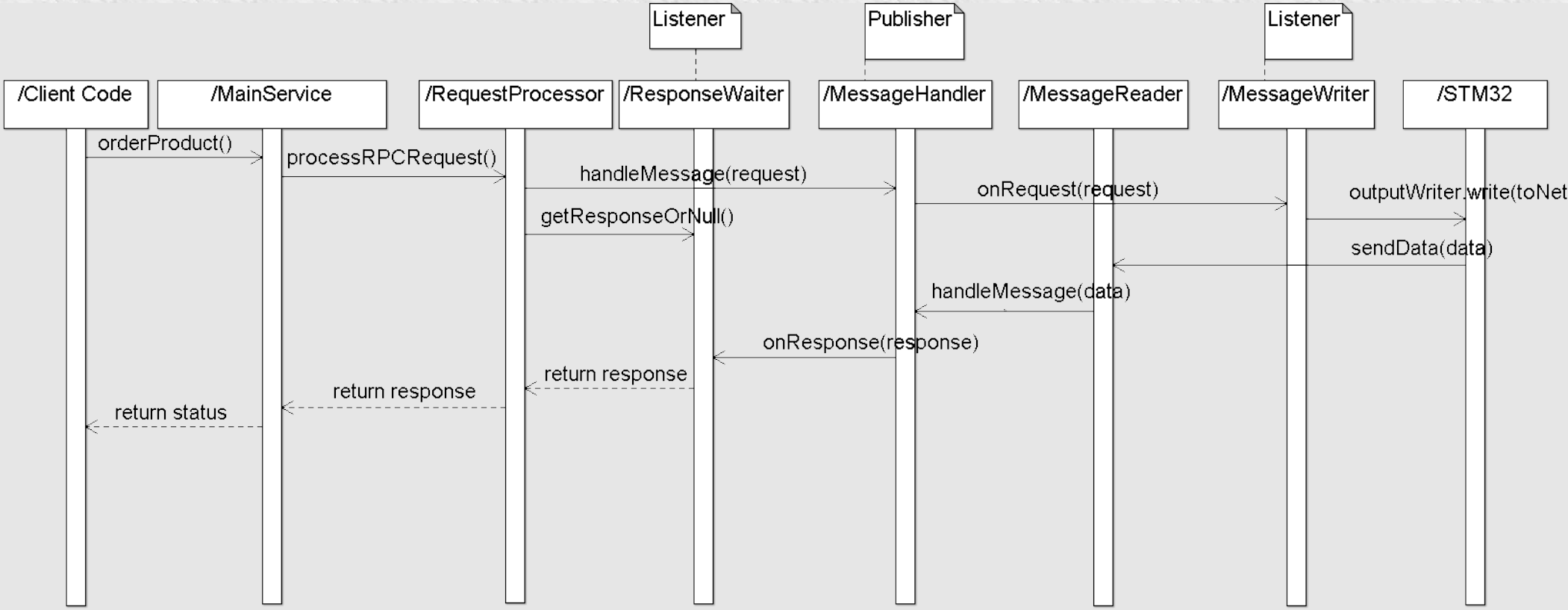
Server application architecture



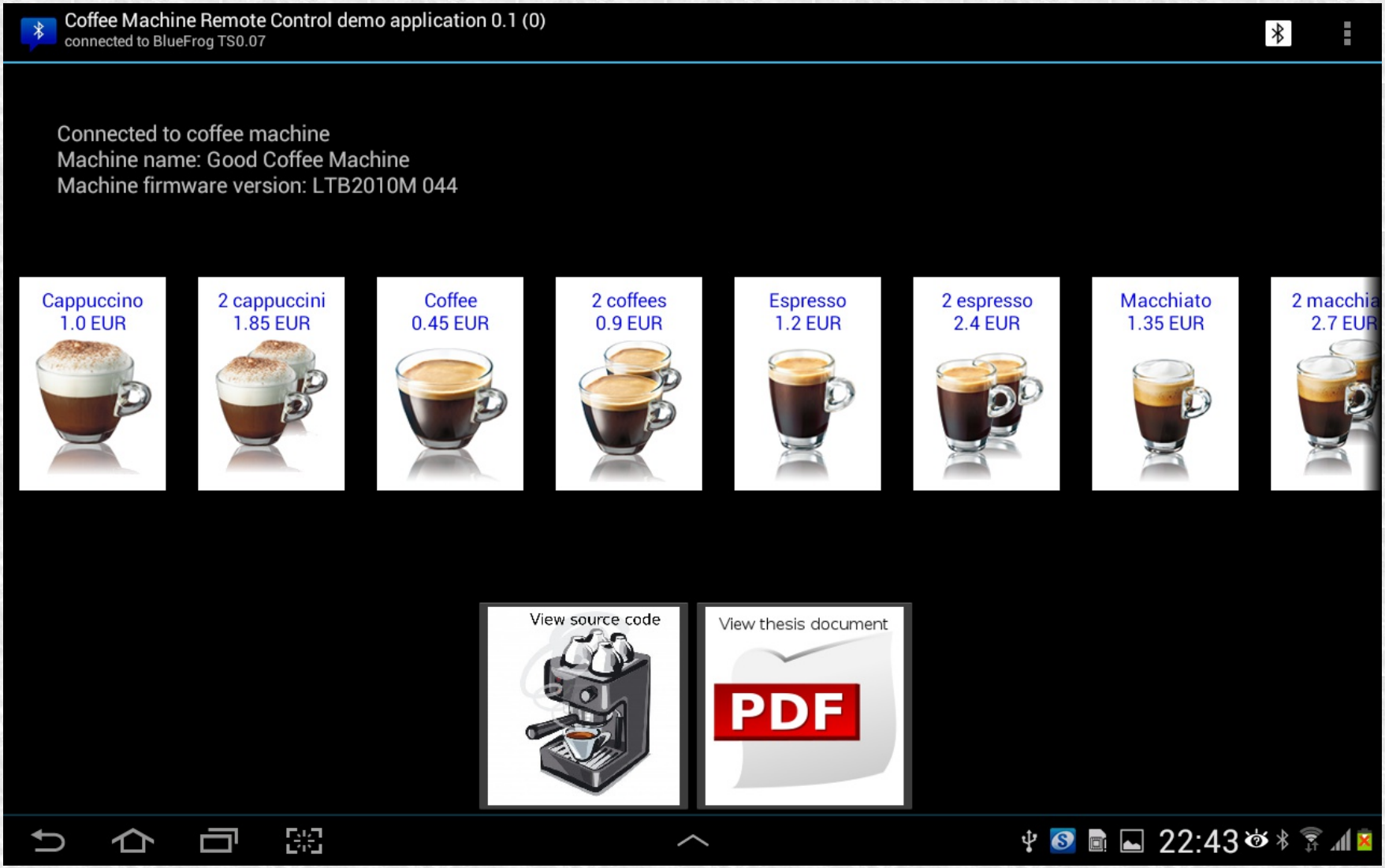
Client Java library

```
public interface Service {

    // connect methods
    void connect(Reader inputReader, Writer outputWriter);
    boolean isConnected();
    void disconnect();

    public void start();
    public void stop();
    public boolean isRunning();

    boolean addListener(RPCServiceListener listener);
    boolean removeListener(RPCServiceListener listener);

    void setTimeoutMs(long timeoutMs);
    long getTimeoutMs();

    void setRequestProcessor(RequestProcessor requestProcessor);
    RequestProcessor getRequestProcessorByType(
            Class requestProcessorClass,
            Object... params);
}

public interface CoffeeMachineService extends Service {
    ServiceContract getServiceContract();

    Map<String, Object> getInfo();

    // products
    List<Product> getProducts();
    Product.Status orderProduct(int productId);
    Product.Status cancelProduct(int productId);
    Product.Status getProductStatus(int productId);
}
```

Client Java library



Android client application



DEMO

Thank you!