NORTHERN ARIZONA UNIVERSITY
College of Engineering, Forestry, and Natural Sciences

Cloud Based API for On-Site Hotel Premises

Team: P. Patel, A. Ramirez, S. Strickland, J. Serafin
Mentor: Ana Paula Chaves
Sponsor: Tony Pallas at SkyTouch Technology

SkyTouch TECHNOLOGY

Motivation

SkyTouch Technology is software as a service.

- They are partnered with Choice Hotels; SkyTouch provides them with different hotel software.
- SkyTouch provides software to more than 5000 hotels. There are a lot of data gets transferred around within these software.
- some of these subsystems support modern TCP/IP-based communication, there are still many that rely on legacy RS-232 serial port connections to send and receive information.
- To handle legacy communication, current solutions require custom software to be physically hosted at a hotel.

Problem:

Developing custom software tailored to each hotel is costly to create, and even more costly to maintain over time. Team Cloud Connect is aimed at improving the capability of SkyTouch Hotel OS to access and manage the many varied hardware subsystems installed in hotels.

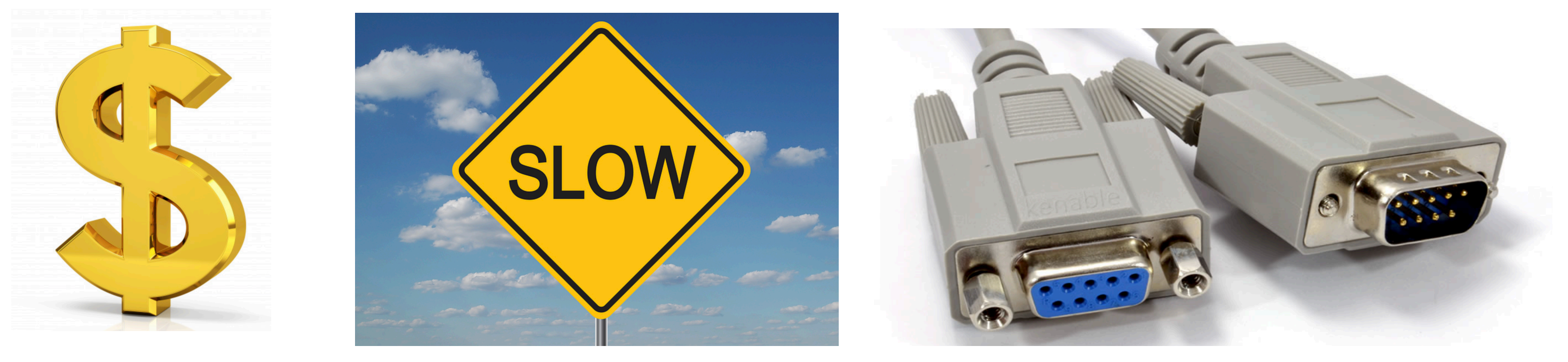


Figure 1: Problem Visual

Solution Overview

Our custom Java application (VADER) handles XML data parsing for different devices. It will be easily be resalable for any other hotel interfaces that SkyTouch have in future. This solution will eliminate the need for a custom application to be deployed within each hotel. By doing so, we will avoid RS-232 serial port connection and customized software for each hotel , saving money and time.

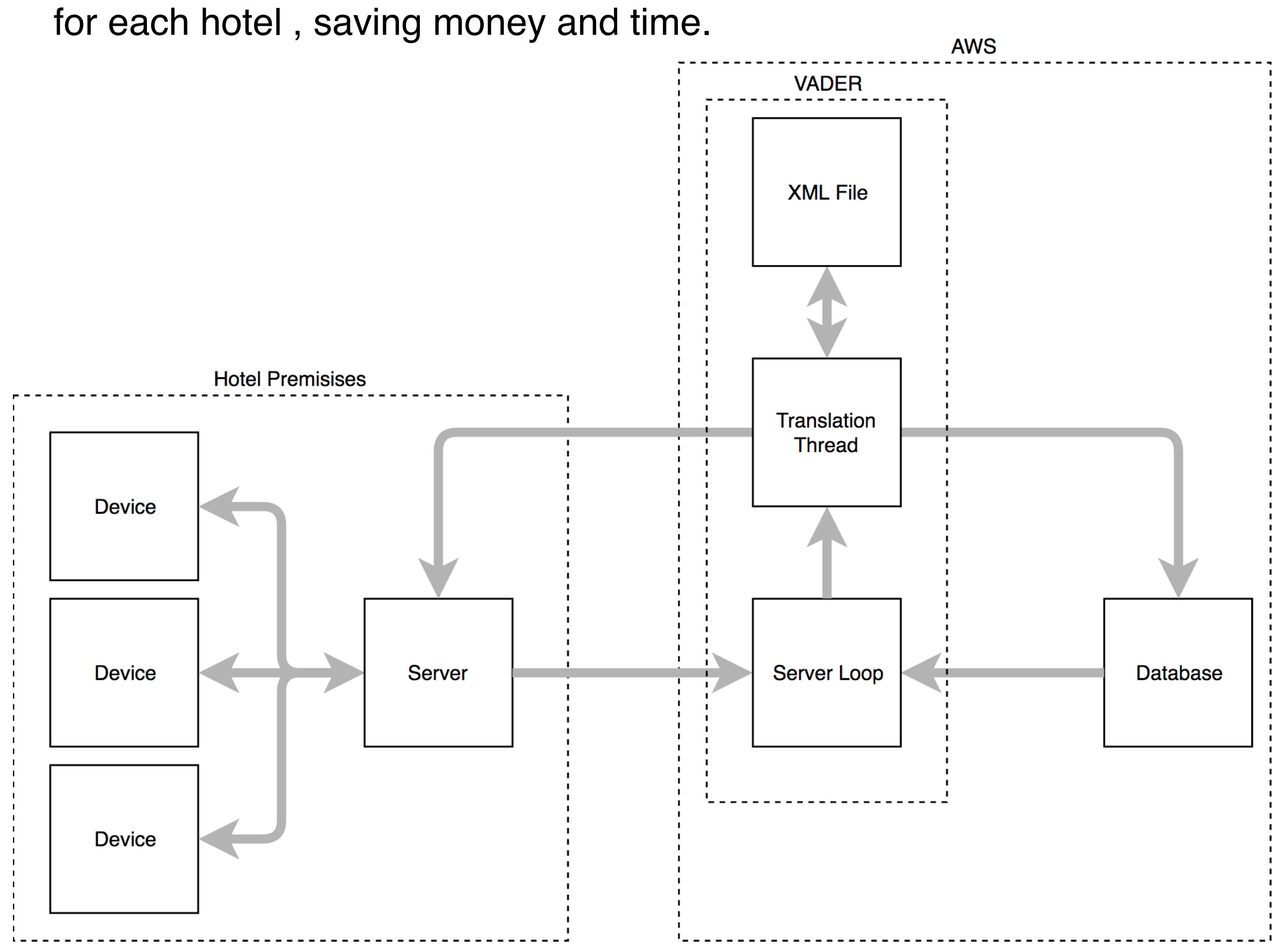
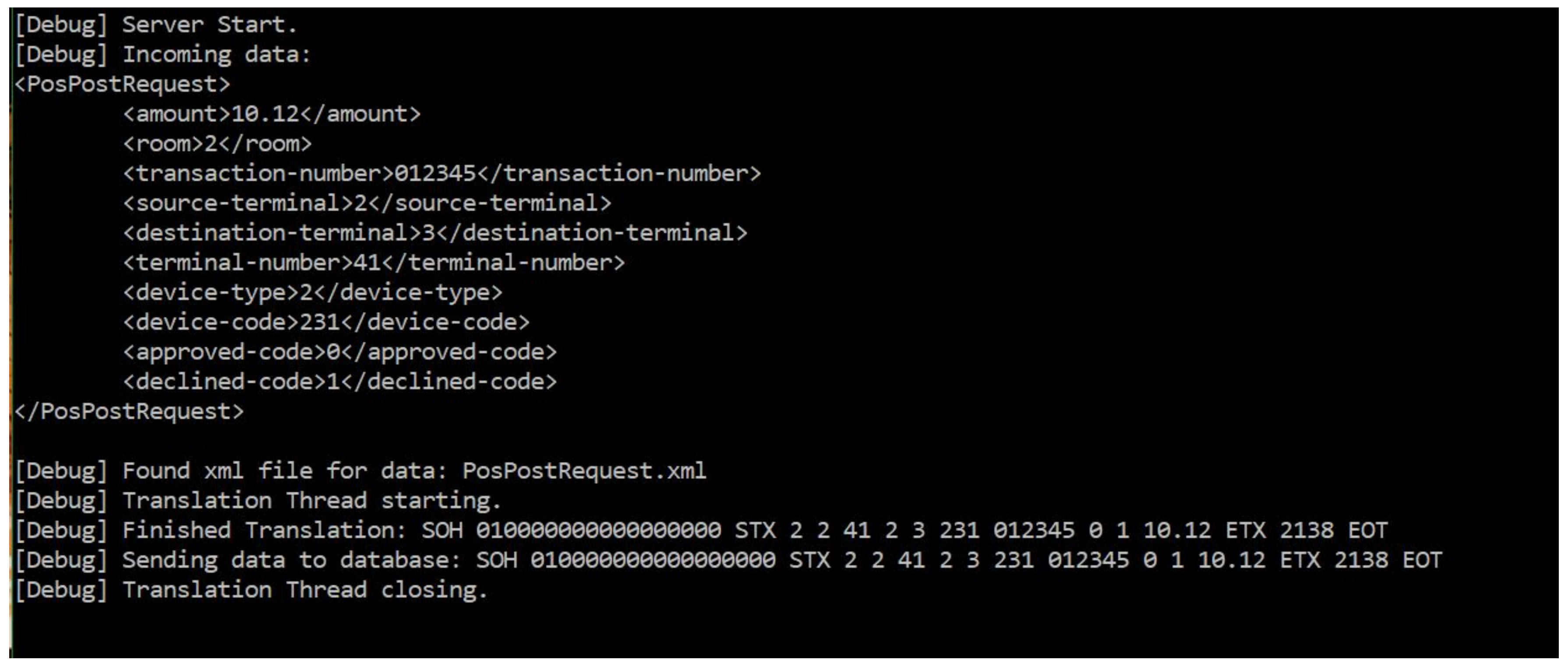


Figure 2: Overview of VADER

Key Features

Some Key Features of our solution are:

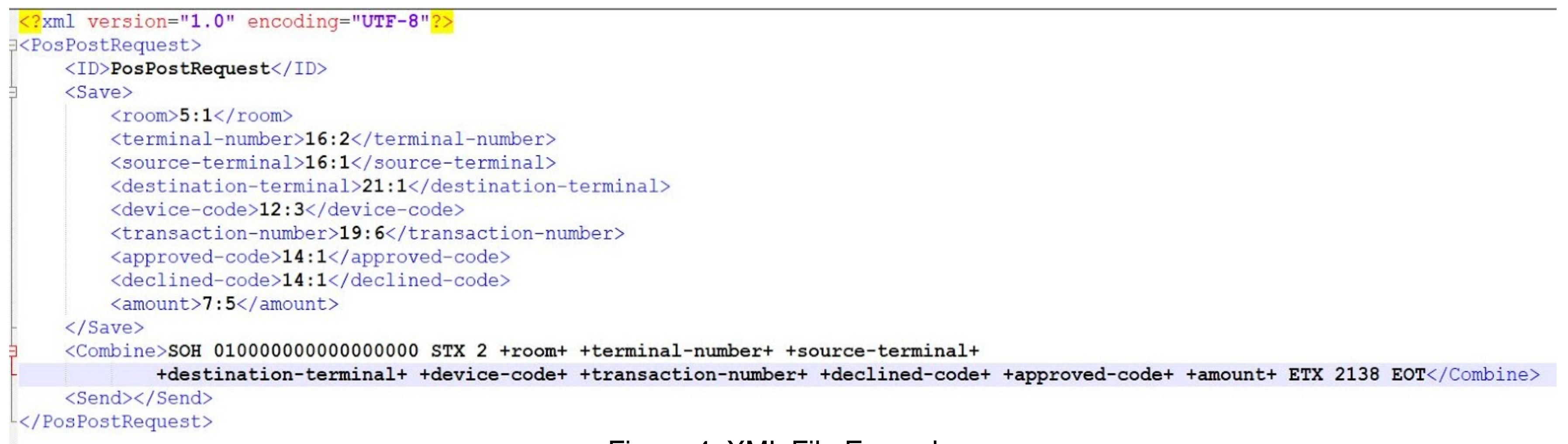
- **Server Functionality:**
  Vader runs on a server inside of Amazon Web Services (AWS), using multithreading and complicated conditionals, Vader can handle as many concurrent solutions as SkyTouch Technologies needs, all they need to do is give the server more resources, and the server can handle more and more connections.
- **XML File:**
  The use of an XML file to dictate to the program how to do the translation it needs makes our solution very flexible. It allows SkyTouch Technologies to create their own files and use them, rather than having to reprogram a new solution anytime a new device gets added or a problem is found.



```
[Debug] Server Start.
[Debug] Incoming data:
<PosPostRequest>
        <amount>10.12</amount>
        <room>2</room>
        <transaction-number>012345</transaction-number>
        <source-terminal>2</source-terminal>
        <destination-terminal>3</destination-terminal>
        <terminal-number>41</terminal-number>
        <device-type>2</device-type>
        <device-code>231</device-code>
        <approved-code>0</approved-code>
        <declined-code>1</declined-code>
</PosPostRequest>

[Debug] Found xml file for data: PosPostRequest.xml
[Debug] Translation Thread starting.
[Debug] Finished Translation: SOH 010000000000000000 STX 2 2 41 2 3 231 012345 0 1 10.12 ETX 2138 EOT
[Debug] Sending data to database: SOH 010000000000000000 STX 2 2 41 2 3 231 012345 0 1 10.12 ETX 2138 EOT
[Debug] Translation Thread closing.
```

Figure 3: Server Functionality



```xml
<?xml version="1.0" encoding="UTF-8"?>
<PosPostRequest>
    <ID>PosPostRequest</ID>
    <Save>
        <room>5:1</room>
        <terminal-number>16:2</terminal-number>
        <source-terminal>16:1</source-terminal>
        <destination-terminal>21:1</destination-terminal>
        <device-code>12:3</device-code>
        <transaction-number>19:6</transaction-number>
        <approved-code>14:1</approved-code>
        <declined-code>14:1</declined-code>
        <amount>7:5</amount>
    </Save>
    <Combine>SOH 010000000000000000 STX 2 +room+ +terminal-number+ +source-terminal+
            +destination-terminal+ +device-code+ +transaction-number+ +declined-code+ +approved-code+ +amount+ ETX 2138 EOT</Combine>
    <Send></Send>
</PosPostRequest>
```

Figure 4: XML File Example

Technologies

Figure 5: Technologies

Challenges

Figure 6: Obstacles during development

Future Work

Improvements to our VADER solution include moving the serial converter to a mobile platform. This way, hotels can have the option to move all on site devices to send information to another mobile device. VADER can then be moved to a mobile app where the translation can occur and be sent onto a SkyTouch database hosted on Amazon Web Services.

Figure 7: Future work

Testing

We had two situations that we needed to test to confirm full functionality. Our first test case was sending a message from a program simulating a device in a hotel through VADER and into a database simulating the current SkyTouch database. Our second test case was the inverse. We had to send a message from the simulated database through VADER and to a simulated device. Both test cases had to work with, and without, a proper XML file dictating how to translate the message.

Architecture

VADER, the new architecture implementation is shown on the left and the XML file format architecture is shown on the right.

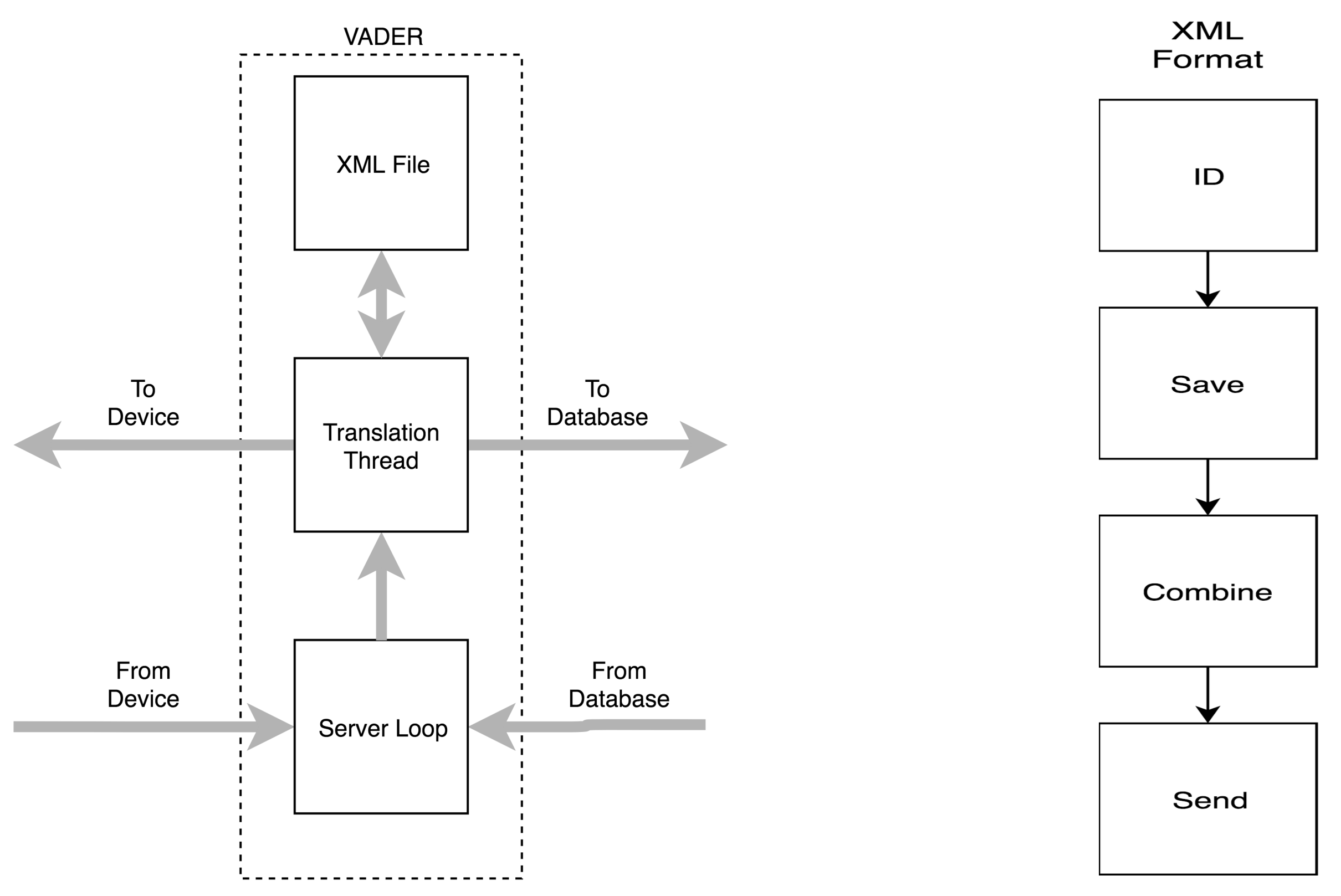


Figure 8: Architecture overview

The Vader Architecture includes three new modules:

- **XML file** – XML File Structured in such a way Translation Thread can process
- **Translation thread** – Processes any incoming XML files accordingly
- **Server loop** – Listens for any incoming connections and sends them to translation thread

XML File format includes:

- **ID** - Contains String to specify if file is valid or not
- **Save** – Contains instructions on what to save
- **Combine** – Combines the instructions in save field
- **Send** – Sends message accordingly

Outcome

- Save the company money
- Efficiency
- Faster Results
- Remove need for individual installations at hotels