Distribution and Integration Technologies (TDIN)

Assignment # 2

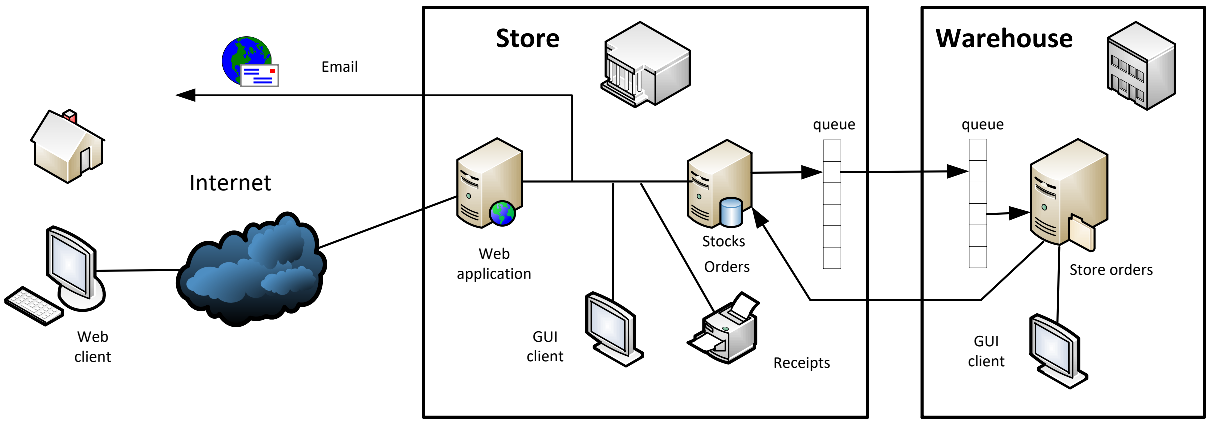
An enterprise distributed system

2019

Scenario

A book editor prints and sells books in his own installations and book shop. It intends to develop a system for coordinating its sells, orders and stock management. The editor owns two facilities physically different, the bookshop store with a public exposition area, and a warehouse for storing larger quantities of books. The editor intends also to make available a web application for remote consulting and ordering.

In the bookshop store a web server is running, hosting the web application, and some persistent record of available titles, price and existences (stock) in the bookshop and warehouse. Some services can also be hosted in the same server. This server is **always on** and is connected to the internet. Between the store and warehouse there is also a network connection, but the warehouse computers usually run only in labor hours, including its own server.



In the store and warehouse there are also GUI clients operated by the workers of those places.

The main requirements and operation of the all system are as follows:

- The store web server accepts orders from the internet, using a web application. Each order should specify a title (for simplicity assume only a title per order/sell), quantity, client name, address and email. For each order, a unique identifier (GUID/UUID) is also generated. All created orders are stored on this server and have a state. For simplicity, we can ignore the payment process in this scenario, and consider only a few book titles available (but try to be realistic).
- The store GUI application, operated by some employee, allows selling titles existing on the store for a visiting client, or the creation of an order (identical to a web order) if the title only exists on the warehouse. A sell specifies also the client name, book title, quantity and total price, updates the store existence (stock) and prints a receipt (use a separate application (console or GUI) to simulate the printer, where the receipt will appear). When a title doesn't exist in the store, an order should be created.
- Orders (originating from the web or the GUI store application) can be in one of three states: "waiting expedition", "dispatched at … (date)", "dispatch will occur at … (date)". When an order is created, an email should be sent to the client with the details (title,

quantity, price per book, total price paid). If there is stock in the store the email should contain the indication “dispatched at … (next day date)”, otherwise it should indicate “waiting expedition”.

- All new orders are recorded in the store server. This server verifies the store stock. If the stock is enough, the server updates it and considers the order satisfied in the next day. If the stock is not enough a request (a service call, using a message queue) is sent to the warehouse server, for a quantity of 10 books plus the initial order volume. In this case the initial order state should be “waiting expedition”.
- The warehouse server receives these calls asynchronously through a message queue (linked to a service) and persists locally its data for ulterior consultation and processing.
- The requests to the warehouse can be consulted and manipulated by a GUI client application in the warehouse. When the warehouse is about to ship the title to the store (indicated by an employee action in the warehouse GUI client), the store server should be notified (since the store server is always on, this could be a simple remote call) and the order state (in the store) should change to “dispatch should occur at … (today plus 2 days)”. The books will be now physically transported to the store. Assume that the warehouse has always the number of books requested.
- When the store receives the requested books from the warehouse, a clerk must accept them in the store GUI application, causing an update to the store stock and changing the state (to “dispatched at ... (today’s date)”) of all pending orders that can be satisfied. In this case a new email should also be sent to the client.
- Remote clients, through the web application, can consult the state of their orders at any time.

Technologies

Implement this distributed application following the Service Oriented Architecture (SOA) principles. You can choose any appropriate technologies, but they need to include remote services (XML or REST), a queue for not loosing requests to the warehouse, and a dual service (or equivalent) for ‘calling’ the printer (which is simulated by a client application) from the service at the store.

The service at the store server should support several operations like:

- consult the store stock of a title
- make a sell (updates the store stock and prints a receipt)
- create an order
- change the state of an order
- any other convenient operations

The service at the warehouse should support operations like:

- get requests that arrived from the store
- complete a request
- any other convenient operations

The information needed to the servers in the application can be persisted in files or local databases.

Realization

The user interface should allow the realization of the specified requirements in a comfortable way. For testing and demonstration all services, servers and applications can run on the same computer. You can complement the specified operations with any other functionalities considered useful or relevant.

The satisfaction of all requirements, ease of use, addition of other operation functionalities, and good practices and architecture, are factors taken into consideration for grading.

Report

You should write a small report containing a detailed architecture specification (the composing modules and services and their interaction), the functionalities included, any testing done, and a graphical representation (screen captures) of the main flows of use.

You should state also all the conditions and instructions to build and run your applications.