11

Assignment 02: Software Architecture Document

Date: November 20, 2022

Course: PA1453

Group 9

11

12

Table of contents

Overview

Documentation Roadmap

As we know, this assignment is a continuation of assignment one. So, regarding the system specification document (Assignment 1), OTIS system supposes to provide direct access for Lycia's customers to order reports and minimize internal employee involvement in report generation. Design Purpose, Primary Functionality, Quality Attributes, Architectural Concerns, and Constraints have been completely covered in the previous assignment. Now, it is time to step forward in creating and documenting software architectures. Besides the overview section, this

21

13

document consists of two different views including Module View and Component & Connector (C&C) view. By Going through each of those views, you can find below topics:

A graphical presentation that illustrates the view's elements and relations is provided in **Primary Presentation.**

Elements' properties and the relations between the elements are described in **Element Catalogue**. Also, we have tried to show the element behavior by drawing UML diagrams.

Variation points throughout the architecture that are used in this view, are depicted in **Variability Guide**.

Our architectural decisions and rationales behind those decisions are provided in the **Rationale** section.

The only part which is updated from our previous file is **Context Diagram.** The external systems and actors are the same in both views.

Views Overview

The most important concept associated with software architecture documentation is view. As we know, a view is a representation of a set of system elements and relations among them, not all system elements but those of a particular type. Also, different views will highlight different system elements and relations. Traditionally there are three important categories of architectural structures including Module Structures, Component and Connector (C&C) structures, and Allocation structures.

Here, two of them are shown (Module Structures, Component and Connector (C&C) structures). For presenting the module view, our team has decided to choose *layers style* to provide abstraction. It consists of four layers (Presentation, Service, Business, Data), an External Systems package, and relations among them *(Figure 1)*. Besides, you can find a *Keywords Explanations table* to comprehend the provided presentation better.

On the next section, you can find Component-and-connector (C&C) structures. For C&C we chose to present a component diagram. This helps better illustrate how the different components are communicating with eachother. Since for c&c we do not need to know how a component is built, any such information here would only overcomplicate the view and clarity would be decreased.

Kommenterad [fb1]: I covered (brief overview) and short description about (module view). can you add a little about c&c ?

Kommenterad [fb2R1]: @martin

Design Decisions and Rationale

31

14

We have chosen to design using microservices, since the dev team for OTIS are proficient at using Java Spring Boot. Which has many features that make building and running microservices easier. This will also improve upon the most important quality attribute: availability (most important for most, if not all, stakeholders). Hosting the different services in multiple regions where there are many users would ensure a much lower downtime of the system while also increasing performance since a single server/connection would not take care of all traffic/computation.

Kommenterad [ML3]: Honestly I am pretty sure we have designed a set of monolithic architectures. But I think we can keep the parts where I mention microservices until after we have spoken to the assistant.
If it turns out we are using a monolithic system with regular services I will rewrite/remove those parts.

Kommenterad [VS4R3]: We already stated that we are using micro services. Also we added Orchestrator, which will ensure availability for our containerised infrastructure.

Kommenterad [ML5R3]: forgot to remove this comment after the meeting with Muhammed. my bad :)

Directory

Glossary

API – Application programming interface. One of the ways components and/or devices can communicate (send/request data).

C&C – Component and Connector. A fancy way of showing how components are connected and what type of connection is used during runtime. For example, a users device would connect to the web interface with a https connection.

Modular – Similar to c&c, but with a heavier focus on how the components are built and how they depend on eachother.

Performance – How responsive a system is. For example, if a user clicks on a “user profile” button, it should never take more than 1-2 seconds before the page of a user profile is shown on screen.

Availability – How often the system is available to the user, preferably there should be as little unavailability as possible.

Usability – How easy the system is to use/how long it takes to learn how to use the system.

Security – How difficult it would be for an unauthorized user to get his/her hands on any secret data and/or deny access for authorized users to the system.

Kommenterad [ML6]: Personally I would prefer having indexes/glossaries at the end.
But if usman wants a glossary in the beginning/middle of the document, his will shall be done.

Kommenterad [VS7R6]: I agree.

View Documentation (Modular)

Primary Presentation

41

15

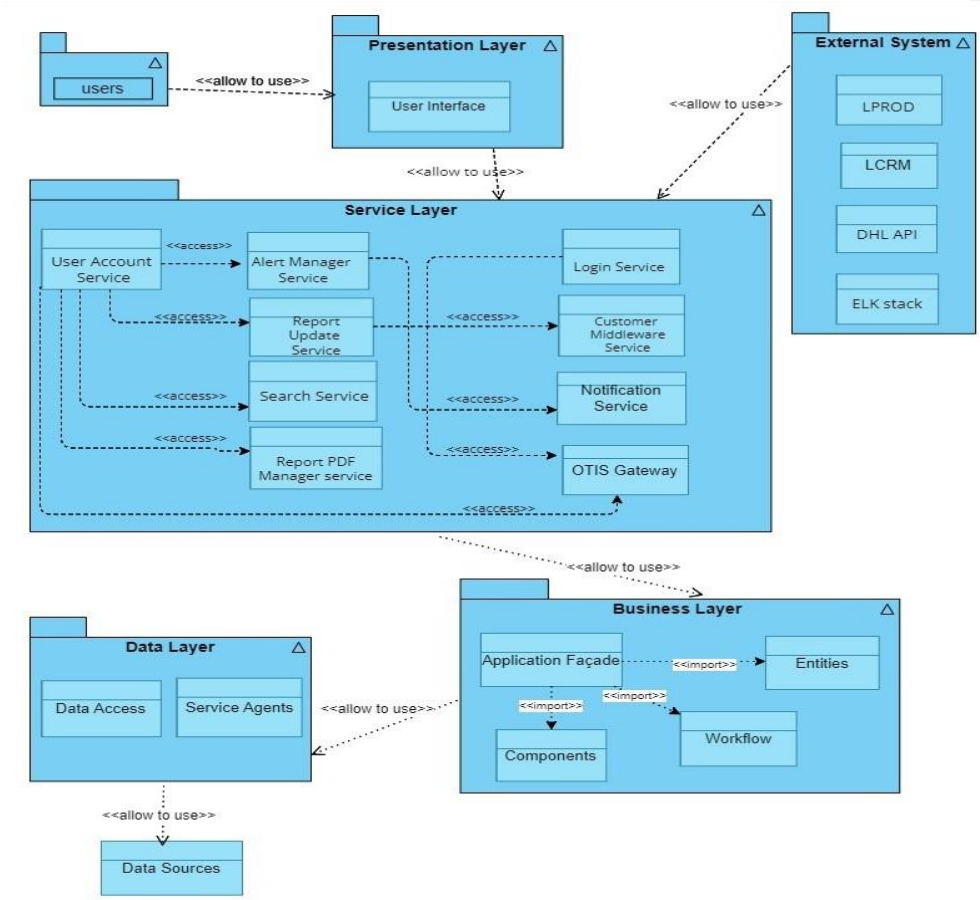


Figure 1: Modular view (Primary Presentation)

Keywords Explanations
«allow to use» indicates that module can interact with another module
«import» indicates public package import
«access» indicates private package import

51

16

A folder icon with a small triangle in the upper right corner of the large rectangle notates using the ordinary package symbol
A rectangle with a line in middle of itself is used to show a package

Table 1: keyword explanation

Element Catalogue

There are five layers among the system elements including:

- Presentation Layer
- Service Layer
- Business Layer
- Data Layer
- External Systems

And each layer has several elements which are described below:

Presentation Layer

All system users will see their access points via User interface façade.

User Interface

Each user needs a gateway to connect to the system that can be a mobile application or a web-based application (PWA) to link to related data regarding their purposes. All Services (described in Service Layer) are accessible for humanistic users and agents via OTIS User Interface module.

Service Layer

OTIS need to be service-based application to cover the majority of current requirements for gathering data and generating reports for different kind of clients (mobile application, web-based browsing on PC, etc.) and with current situation of system we have recognized 9 basic services to cover the expected functionality including:

1. OTIS Gateway
2. Log in Service
3. User Account Service
4. Alert Manager Service
5. Notification Service
6. Report Update Service
7. Report PDF Manager Service
8. Customer Middleware Service
9. Search Service

61

17

OTIS Gateway

This is a core service to handle a variety of inputs (Text, File, Media, etc.) and illustrate data on screen. There are 4 different UI pages including:

1. Login
2. User profile
3. Alert manager
4. Search

Which can only be reached after successful authentication. The first page shown will always be the login page, where a user will input username/email and password which will then be sent to the Login service. Upon unsuccessful login an ambiguous error message could be shown, as to not give away any secret information. When the user has cleared the authentication process the page could be swapped to the Search module and we can swap freely between the 4 user specific pages or log out.

Log in Service

Uses user input from the web interface and sanitizes it and checks for invalid input. If the input is valid, it is then checked against a username stored in the user information DB. Upon successful login we are authorized to use the User Account Service. Valid input, invalid input, successful authorization and unsuccessful authorization is sent to the Logger service. Also, IP-addresses will be logged on too many failed attempts.

User Account Service

This service shows the current users account information which can be edited and a list of all orders, where we can choose an order to get a more detailed view. Also, it can send update requests and receive updates with the Report Update Service.

Alert Manager Service

There 2 different messaging policy in OTIS: push notification and pull-based messaging. This service will cover pull-based mechanism to send alerts/notifications to their desired device depending on changes/updates in their order details. If an event that is marked for alerts gets an update, information should be sent from Alert manager service which would then be sent as an alert/notification to the user's device.

Notification Service

This service will cover a push notification mechanism to send alerts/notifications to different services to feed them for further actions regarding their triggers and functionalities. The Logging service and synchronization process are fundamental consumers of this service across total events regarding to different modules.

Report Update Service

If changes are made to the Reports database this service makes sure all details and information are updated. This service communicates with user account information and the report database. Also, it communicates with customer middleware service to demand a current update when user asks for it.

Report PDF Manager Service

This service puts data from report DB into a more readable format for download and viewing (PDF files). It communicates with Report DB, and File Storage (SFTP) modules as well.

71

18

Customer Middleware Service

This service gathers data from the external system and stores them in the report DB. It uses a push API provided by DHL customer middleware subscribes to get automatic updates and then stores these in the Report DB. Also, it updates LCRM with new customer information and should handle updates both on demand and once an hour.

Search Service

This service provides facilities to search entire system based on accesses that allocated to each user. It connected to User Account Service to consider users permissions regard to the different part of the system. With this service, users can search for historical data (including PDF reports) instead build them every time.

Business Layer

OTIS should consider a different layer of modules regarding next development points and usability and performance of the system. There are 4 different modules here including:

1. Application Façade
2. OTIS Workflow
3. Components
4. Entities

Application Façade

Application Façade is responsible to handle interactions with the domain model and getting the information to the service layer endpoints in exactly the form that the they require. In this pattern each components needs to know nothing about what is going on in the model, it only handles the functionality.

OTIS Workflow

OTIS Workflow is the core engine of the system that handles interactions between different components and modules. There are three main components in OTIS workflows including:

1. External Service Brokerage
2. Notifier

Each workflow in OTIS has its Components and Entities described below

External Service Brokerage

External Service Brokerage is a workflow to handle interactions with external systems (LPROD, LCRM, DHL). Messages are the base entity which works with the other components. The next component here is the scheduler that handle interaction with different external system based on regulatable timing algorithm.

Notifier

Notifier is responsible to create messages and manage it in all part of the system. The related components regard to messages are:

1. Generating Message
2. Sending Message
3. Receiving Message

81

4. Finding Message
5. Scheduling

Other Components and Entities

Variant users in OTIS with different capabilities are handled with defining different entities in the System Including:

1. Person (an agent with general attributes like name, address and so on)
2. User (a person who registered in the system and has own type, username and password)
3. User Type (Clients, DHL, Lycia, Account Manager, Admin)
4. Permissions

Data Layer

Data Layer is located at the deepest point of the OTIS. It has ORM levels access to databases and the other resources and all class files are designed to return the result of queries that each component at business layer asked for.

Data Access

In this part there are same classes and entities with business layer's one to interact with internal calls from the above layer which mean these entities need to be existed in implementation:

1. Message
2. User

Service Agent

Service Agent is a part of the Data Layer which accessed physically to data resources to apply changes to them. We have determined three different resources including:

1. User Information DB
2. Report DB
3. File Storage

External Systems

External Systems are cross cutting part of the system which need to be integrated with external parties including:

1. LPROD
2. LCRM
3. DHL API
4. ELK STACK

LPROD

An external endpoint that produces information about the quality of the product and information related to the process of creation of said product. Also, it communicates with Customer Middleware Service.

LCRM

110

An external system that handles customer information. It can send and receive information to/from the Customer Middleware Service.

DHL API

This endpoint pushes API that will automatically send information about subscribed orders to the Customer Middleware Service.

ELK STACK

This external system is an aspect regarding all logs entire system. The main purpose of this feature is providing different ways to logging events and important data. It has advanced tools behind (including Kibana, Logstash and Elastic Search) to generate and store logs accurately and quickly as much as can be done in the system.

Context Diagram

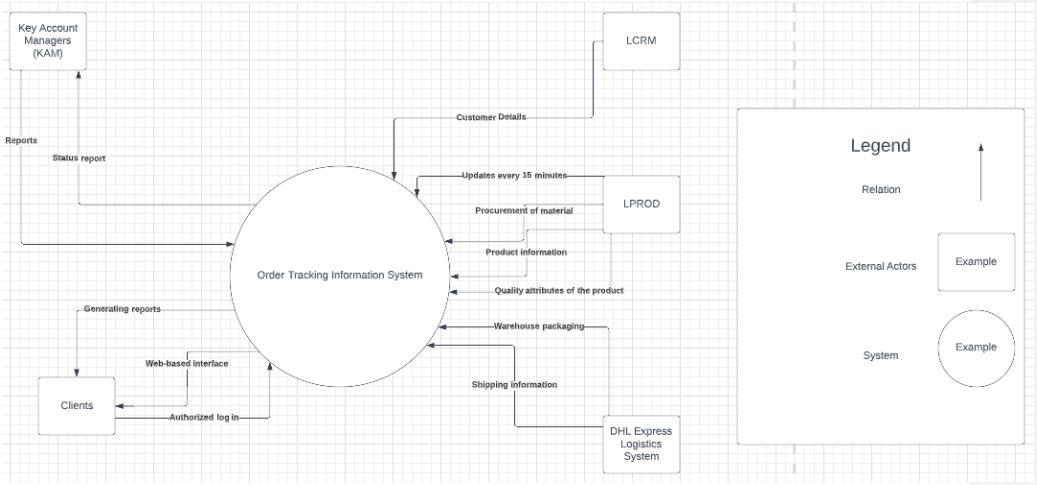


Figure 2. Context diagram of the OTIS system and it's external systems.

Variability Guide

101

Services

The Elk system could change depending on future plans and for other efficiency attributes. The User interface could change depending on ease of use for the customer. Different notifiers could be changed to notify the customer the appropriate way for each customer.

LCRM format

The current format used by the OTIS system is in XML format. This format should be updated the next three months to a JSON supported system.

LPROD Quality updates

To increase the quality from the products updates in the quality measurements are required. Therefore, continuous updates about quality information will be implemented.

Rationale

Tactics selected and justification for each QA

Kommenterad [ML8]: Add QA tactics for why elements where added.

Kommenterad [ML9R8]: A load balancer, if added, could fall under the "control resource demand" performance tactic or the "prevention" availability tactic for example.

Availability

All services are using the fault detection tactic by sending error logs to the logger service. Which will then help admins analyze them and figure out why they are happening.
The OTIS gateway is using a version of the recovery tactic and prevention by “removal from service”. It can redirect a users requests to a different microservice instance if the previous one is failing to meet demands. The faulty instance of the microservice can also be temporarily shut down to make sure that there are no faulty services affecting other parts of the system. The business layer will also use the fault detection with timestamping and condition monitoring to make sure that everything is delivered and that the right information is delivered to the customer needs. The data layer will also use heart beat to make sure that the databases in the layer are online and active.

Performance

The manage resources tactic is mainly used to handle the performance demands.
Since we are using microservices we can split the work between many different cloud hosted services and should thus be able to handle a high amount of requests.
Caching of data for the most commonly used data can be implemented in the report update service or customer middleware service which would also improve upon the overall performance.

Security

In the Presentation layer are we using the detect attacks and resist attacks tactics.
Too many incorrect log in attempts would for example trigger logging of that ip address and in the case

112

of an overall increase in incorrect log in attempts by many different IP addresses a warning should be sent to admins so they can analyze what is happening. It is also authentic the user to make sure that the right permissions are given to the correct user. Since we are storing logs for all errors the admins can also look at any sort of attack trends that might be happening. (many incorrect log in attempts at exact hours of the day or within scripts like time delays)
User input is sanitized as to not allow stack overflow attacks or code injections for example.
All secret log in information is encrypted.

Usability
Services are modeled as to handle most (if not all) usability tactics that the dev team chooses to implement. The data layer will also use the aggregate method to group different databases to make it easier for the customer to choose and gather different data from the OTIS system.

Kommenterad [ML10]: There could probably be some more for usability, I am just unable to think of anything that could be mentioned here currently.
Apart from that I believe it looks done.

Other architectural decisions

Microservices

We choose to use microservices to easier being able to update different parts of our program independently of each other. We also prefer the nature of different components being more independent of each other to mitigate issues if some specific components where to fail.

Component Diagram

We choose the use the primary presentation as a component diagram to easier and more briefly see the overview how data traffic will flow in the OTIS system.

- **Availability**

In the Data layer multiple databases will help split the work and makes it easier to keep it up mostly of the time.

- **Security**

To make sure that the right individuals have access to the system security is required.
Therefore, logs will be available to help find any animalities in the system to prevent any damage.

- **Performance**

To split the work between different databases the work will be more manageable and easier to control and make sure that the required amount of work is split and does not overrun the system.

- **Usability**

121

113

To make it easy for the customer it is necessary to make sure that the user interface is easy and have simple alternatives to help the customer find what they look for. One way to help them is to group different values to easier manage and find common attributes t hat usually is preferred.

131

114

View Documentation (C&C)

Primary Presentation

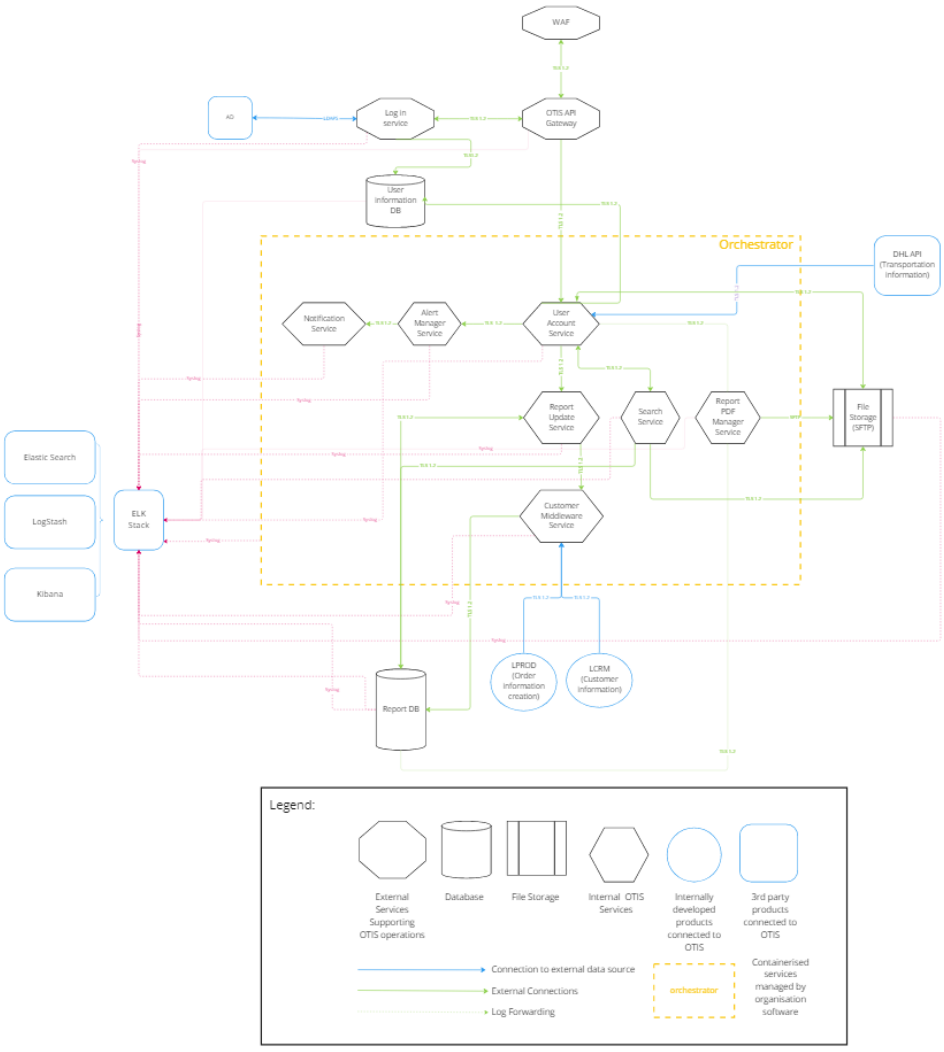


Figure3. c&c primary presentation

141

115

Element Catalogue

UML sequence diagrams

Figure 4: Use Case - Report Generation

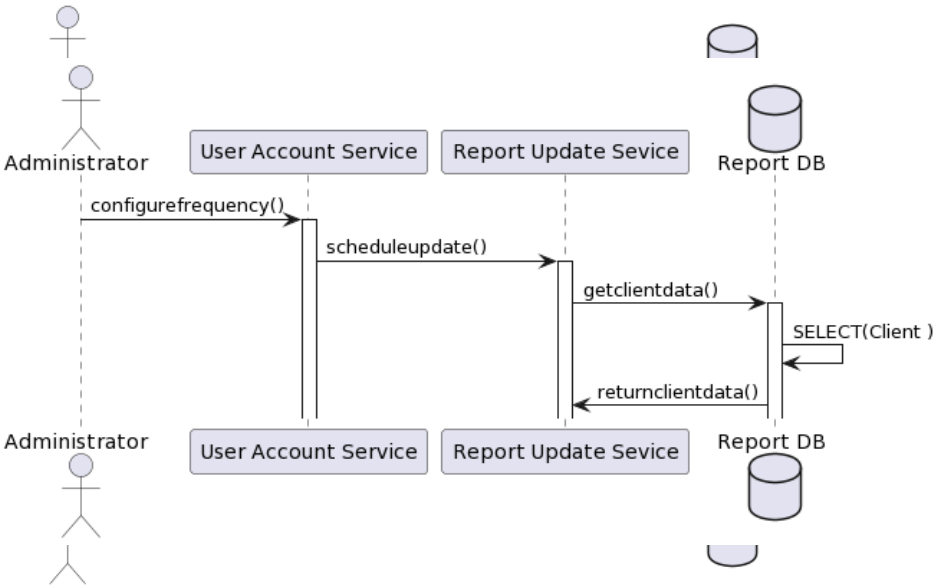
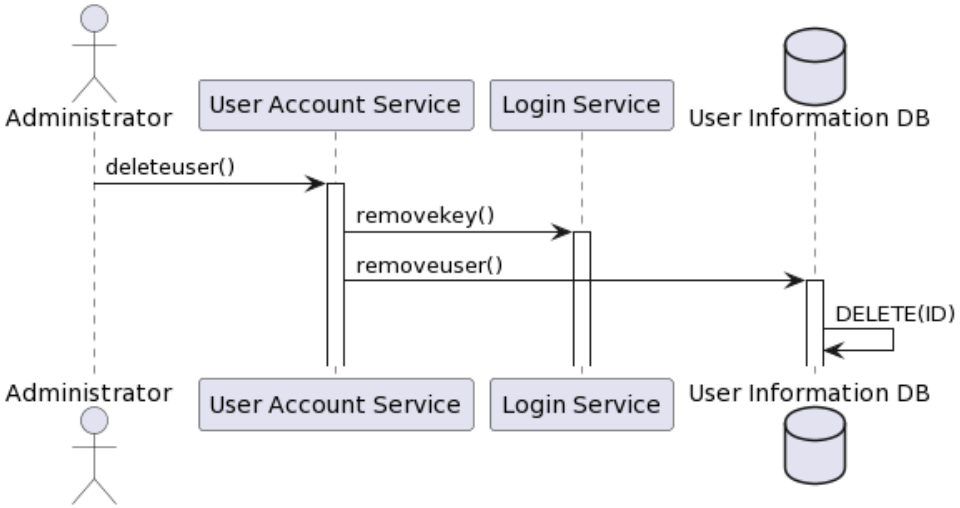


Figure 5: Use Case – Status Update

151

116

Figure 6: Use Case – Manage Users



Notice for all services:
Communication between services will be done via TLS 1.2 connections.
Syslog will be used to send logs to the elk stack.

WAF

This is the component that takes care of user input and showing of data on screen. There are 5 different UI pages in total: Login, User profile, Alert manager, Order details and Order list. (whereas the latter 4 can only be reached after successful authentication) The first page shown will always be the login page, where a user will input username/email and password which will then be sent to the Login service. Upon unsuccessful login an ambiguous error message could be shown, as to not give away any secret information. When the user has cleared the authentication process the page could be swapped to the Order list page and we can swap freely between the 4 user specific pages or log out.

Log in Service

Uses user input from the web interface and sanitizes it and checks for invalid input. If the input is valid, it is then checked against a username stored in the User Information Database. Upon successful login we get an Oath2 token and can interact with User Account Service and/or the active directory (if we have admin rights). Valid input, invalid input, successful authorization and unsuccessful authorization is sent to the Logger service. IP-addresses will be logged on too many failed attempts. (Part of the user agreement policy)

161

117

Also encrypts and decrypts secret user information as to hinder attacks.

OTIS Gateway

Makes sure that user requests are sent to the correct microservice.
Can also handle sending a users requests to a different microservice if the previous one was slow or if it crashed.

User Account Service

Has the current users account information which can be edited.
Has a list of all orders, where we can choose an order to get a more detailed view.
Can send update requests and receive updates with the Report Update Service.
If the Oath2 token has admin authorization we can use admin functions.

Alert Manager Service

This service can be set up by the user to send alerts/notifications to their desired device depending on changes/updates in their order details.
If an order that is marked for alerts gets an update, information should be sent from order details service. Which would then send some information to the notification service (phone number, mail address and/or fax number for example).

Notification Service

Uses the information it gets from the Alert Manager Service to decide what type of notification should be sent and to what device.
Sends a message with a certain level of detail to the customers chosen device. (what types should be discussed with the stakeholders)

Report Update Service

If changes are made to the database this service makes sure that a user's order list, order details and user info is updated. Communicates with order list, order details, user account information and the report database. Should also communicate with customer middleware to demand a current update when user asks for it.

Report PDF Manager Service

171

Kommenterad [VS11]: Should we add Admin interface as a part of User account service? Or as an additional service

Kommenterad [ML12R11]: Either that or we could say that different users have different auth tokens or something.

As a side note while discussing security, should we add some sort of security service that makes sure users are correctly authorized, or could we say that each service has something like this or even the gateway itself?

Like for example an Oath2 service checking connection coming in to the gateway or a similar approach.

Also I remember you mentioning a load balancer, is that something to add in the modular and c&c or only in one of the views?

like this architecture:
https://dzone.com/articles/how-to-achieve-oauth2-security-in-microservices-di

Kommenterad [VS13R11]: OAuth 2 great idea!

Kommenterad [ML14]: I think this would be a variability point, being able to either chose one notification type or multiple etc.
Or if it should be developed by the team/outsourced or use an existing one by API.

Kommenterad [VS15R14]: I've added this point to variability guide.

Kommenterad [ML16]: I'm leaving this vague for now, since we could probably leave this decision in the hands of the stakeholders/users.
Do they want multiple types of notification at the cost of extra development or only the simplest solution?

118

Puts data from report DB into a more readable format for download and viewing.
Communicates with Report DB, Order list and File Storage (SFTP).2

Customer Middleware Service

Gathers data from the external system and stores them in the report DB.
Using a push API provided by DHL customer middleware subscribes to an order and gets automatic updates and then stores these in the Report DB.
Update LCRM with new customer information.
Should handle updates both on demand and once an hour.

Kommenterad [ML17]: Maybe add to concerns later that we are swapping DHL API's to this instead of the query based one.

Kommenterad [VS18R17]: Good idea,

LPROD

External system that produces information about the quality of the product and information related to the process of creation of said product. (furnace and machine output etc.)
Communicates with Customer Middleware Service.

LCRM

External system that handles customer information.
Can send and receive information to/from the Customer Middleware Service.

Kommenterad [ML19]: Maybe we could move the authorization link to LCRM instead of a seperate database? Thinking about it now wouldn't LCRM already have some sort of log in information?

Kommenterad [VS20R19]: I thought LCRM was filled in by Sales team. Not sure if it would be technically possible, but we do could link user account to the email in LCRM.

DHL API

Push API that will automatically send information about subscribed orders to the Customer Middleware Service.
(Does not get called while order is in the production state)

Report DB

Storage for data gathered from external systems.
Gets data from the Customer middleware and sends data to Report update daemon and Report PDF manager.

User information DB

Kommenterad [ML21]: See comment for LCRM

Stores encrypted user log in information.
Communicates with the Log in Service and sends logs to the ELK stack upon database errors.

Logger Service

181

119

ELK Stack consists of Elasticsearch, Logstash, and Kibana:

- Elasticsearch is a search and analytics engine.
- Logstash is a server-side data processing service, which would store logs.
- Kibana lets administrators visualize data with charts and graphs in Elasticsearch.

These services would store all logs from the services and allow administrators to work with logs.

Kommenterad [ML22]: Could we say that admins would get notified upon certain degrees of faults/errors and if too many errors happen within a certain timeframe?

If so that would add to the availability tactics we are using!

Search Service

Makes sure a user can find old reports or search amongst current ones (if they have very many reports).
Gets instructions from User Account Service.
Searches through File Storage and Report DB.
Returns found report to User Account Service.

Orchestrator

Kommenterad [ML23]: @Vitalii

Kommenterad [VS24R23]: Orchestrator is a supporting system. Not a required component for OTIS. But a nice to have service.

Kommenterad [ML25R23]: was more to have some description of it than add it as a service

Orchestrator stands for automated software deployment, management of containers and scaling of application. Orchestrator is a supporting system for OTIS that helps automate infrastructure maintenance and improve availability of the product.

Context Diagram

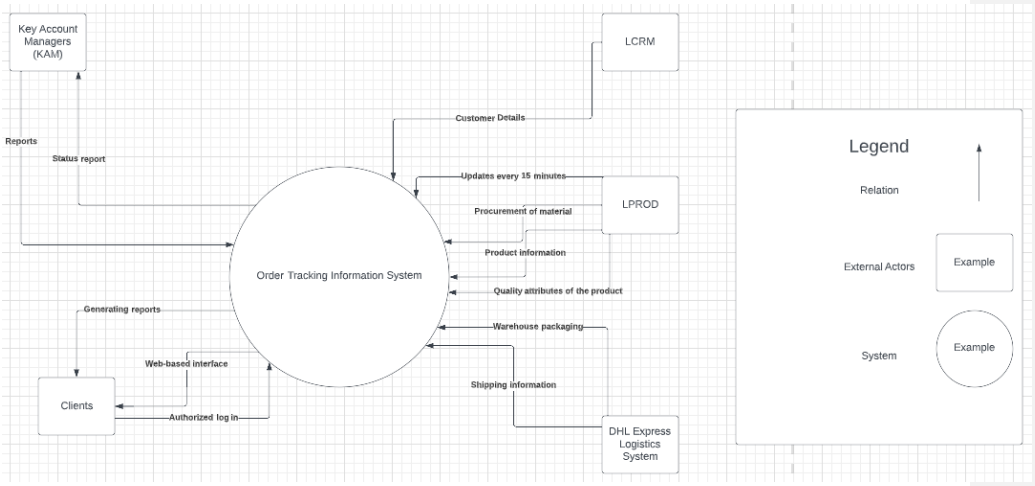


Figure 7. Context diagram of the OTIS system and it's external systems

191

120

Variability Guide

LCRM Export

The current format used by the OTIS system is in XML format. This format should be updated the next three months to a JSON supported system.

Protocols

The current set of protocols used for service communication is the following:

1. The decision was made to use HTTPS where possible (at the minimum of TLS 1.2 for encryption in transit). At the same time for clients that are using old browsers that might be a problem if TLS 1.2 or higher are not supported, so this configuration may be changed.
2. It was decided to use SFTP (Secure FTP) instead of FTP. But this configuration can be changed if needed.
3. It was decided to use Secure LDAP instead of LDAP. But this configuration can be changed if needed.
4. It was decided to use Secure Syslog instead of Syslog. But this configuration can be changed if needed.

Services

The current set of services is the following:

1. The decision was made to use ELK stack as a Logging System, but the ELK stack may be replaced with Splunk or any other data analysis platform if more mature data analysis would be needed by system administrators.
2. Current setup for DHL integration is relying on DHL API that provides delivery details. But DHL has other API that may be useful for application.
3. There are different Authentication mechanisms supported by application: though corporate accounts (for employees – AD) and through application authentication mechanism.
4. Notification service may send alerts using SMS or emails, this setting is available for end users.
5. Report service can provide order status updates with different frequency (1 hour) is a default value, but can be customised as per customer request.

Rationale

Tactics selected and justification for each QA

201

Kommenterad [ML26]: Add QA tactics for why elements where added.

Kommenterad [ML27R26]: A load balancer, if added, could fall under the "control resource demand" performance tactic or the "prevention" availability tactic for example.

121

- Availability
  All services are using the fault detection tactic by sending error logs to the logger service which will then help admins analyze them and figure out why they are happening. The OTIS gateway is using a version of the recovery tactic and prevention by "removal from service". It can redirect a user's requests to a different microservice instance if the previous one is failing to meet demands. The faulty instance of the microservice can also be temporarily shut down to make sure that there are no faulty services affecting other parts of the system.

- Performance
  The manage resources tactic is mainly used to handle the performance demands. Since we are using microservices we can split the work between many different cloud hosted services and should thus be able to handle a high number of requests. Caching of data for the most commonly used data can be implemented in the report update service or customer middleware service which would also improve upon the overall performance.

- Security
  In the log in service, we are using the detect attacks and resist attacks tactics. Too many incorrect log in attempts would for example trigger logging of that IP address and in the case of an overall increase in incorrect log in attempts by many different IP addresses a warning should be sent to admins so they can analyze what is happening. Since we are storing logs for all errors the admins can also look at any sort of attack trends that might be happening. (Many incorrect log in attempts at exact hours of the day or within scripts like time delays)
  User input is sanitized as to not allow stack overflow attacks or code injections for example. All secret log in information is encrypted.

- Usability
  Services are modeled as to handle most (if not all) usability tactics that the dev team chooses to implement.

Kommenterad [ML28]: There could probably be some more for usability, I am just unable to think of anything that could be mentioned here currently.
Apart from that I believe it looks done.

Other architectural decisions

We chose to show the primary view as a component diagram, this lets us have a c&c diagram where we do not show any per-element inside information and can focus on the connections between elements instead. Which will in turn make it clearer how data traffic could look like inside of the system.

QA were addressed as follows:

211

122

- Availability (Use case 1)
  Minimizing the workload by having multiple databases would also make sure that there are less wait time as a cause of one database having to handle all calls/queries. This would also mean that if for example the report database would be down, we could still access the PDF files stored in storage.
  As an overall gain by using microservices we could also make sure that there are many instances of them hosted in multiple regions which would decrease downtime and increase performance even further.

- Performance (Use case 2)
  Splitting the workload between two databases and one filestorage component will make sure that we can minimize the need for multiple connections to a single database, thus improving the response times.
  Having system logs handled by an external system we can save some performance as a cause of not needing to process and store logs within the systems own services and databases.

- Security (Use case 3)
  First of all, we added the most obvious security element, a log in system, so that unauthorized users cannot view/change any of the data they are unqualified for.
  By not letting the users interact directly with any of the databases we can hinder/stop some attacks.

- Usability
  This will more or less be in the hands of the dev team, since we as architects cannot decide for them what a nice looking or easy to use UI would look like.
  We have however made sure that the services for the UI are within its' own service and as such could be modified without having to worry about changes causing problems in other services or the other way around.

Kommenterad [ML29]: @Vitalii
Can we say this?
Asking since to me it kind of looks like we can just send the logs to ELK which then does all the work.

Kommenterad [VS30R29]: Yeah, that's right!

221