Software Architecture

OTIS(Order Tracking & Information System)

Group 9

- Muhammad Mohsin Qamar Khan
- Syed Ali Hasan
- Muawaz Ayyaz
- Ghilmaan Shahzad

Agenda

- Introduction
- Quality attribute and rationale behind their selection
- Architecture concerns and other drivers
- Selected reference architecture and rationales
- Module view - primary presentation and important design concepts (pattern and tactics used)
- C & C View primary presentation and important design concepts (pattern and tactics used)
- Q & A
- End

List of Business Goals and justification

- Give customers an online platform where they can access their order status online all across the world

- Increase the production of the company by reducing the workload on Key Account Managers and Production workers

- Saving audit logs and maintaining the records of the order

- Provide value to customer by providing the state-of-the-art order tracking system

Quality Attribute:

Three quality attributes of the system is detailed below:

- Performance
- Maintainability
- Availability

Rationale for choosing QA: Performance

Performance:

Currently our Manual system provide reports to the customer within 2 months. As our business goal is to reduce the time taken to generate reports to the customers, performance become one of the main quality attribute. Our system should give perform in less time and give customer update about their order status close to near time.

Why?	Customer need to track order records
Sourceofstimulus	Order Tacking
Stimulus	Place new order
Environment	Production / go live
Artifact	CRM portal (for Client)
Response	CRM portal load customer modules
Responsemeasure	View result within 3 second

Rationale for choosing QA: Maintainability

Maintainability :

As our system have different modules, our system should have the ability to add new modules and new functionality within the module. After this build, system could ask to change some functionality in a module or add new module. Every system does through a series of change according to the user needs, so the system has to change. Maintainability is important to add these changes without changing the whole system with less effort.

Why?	Regular data sync required between OTIS,LCRM,LPROD
Source of stimulus	System Developer/ System Administrator
Stimulus	New module integration and upgrading new functionality
Environment	Production/Go Live
Artifact	OTIS, LPROD,LCRM
Response	Ability to make necessary changes
Responsemeasure	System should be available within 20 minutes (Downtime)

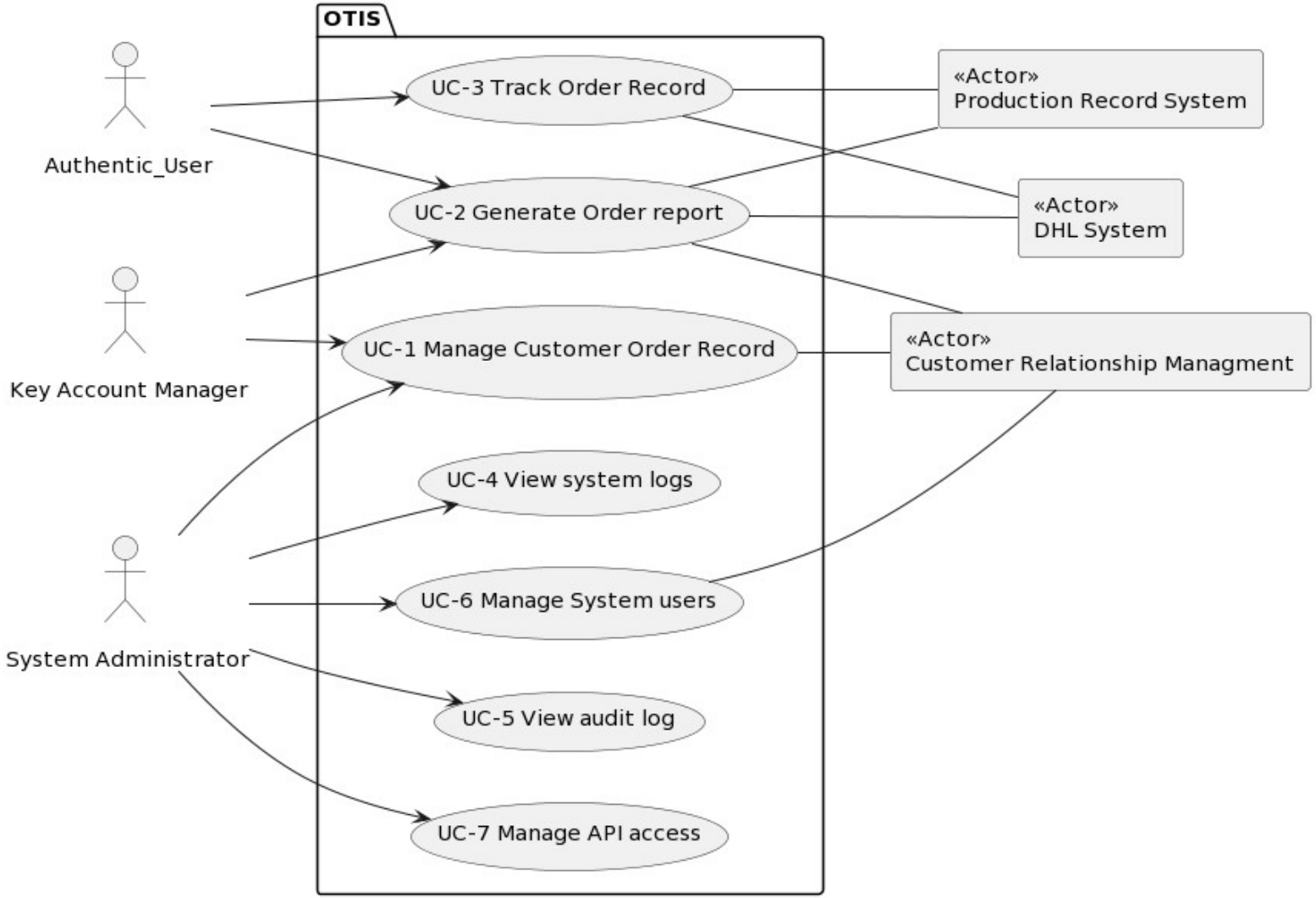
Rationale for choosing QA: Availability

Availability :

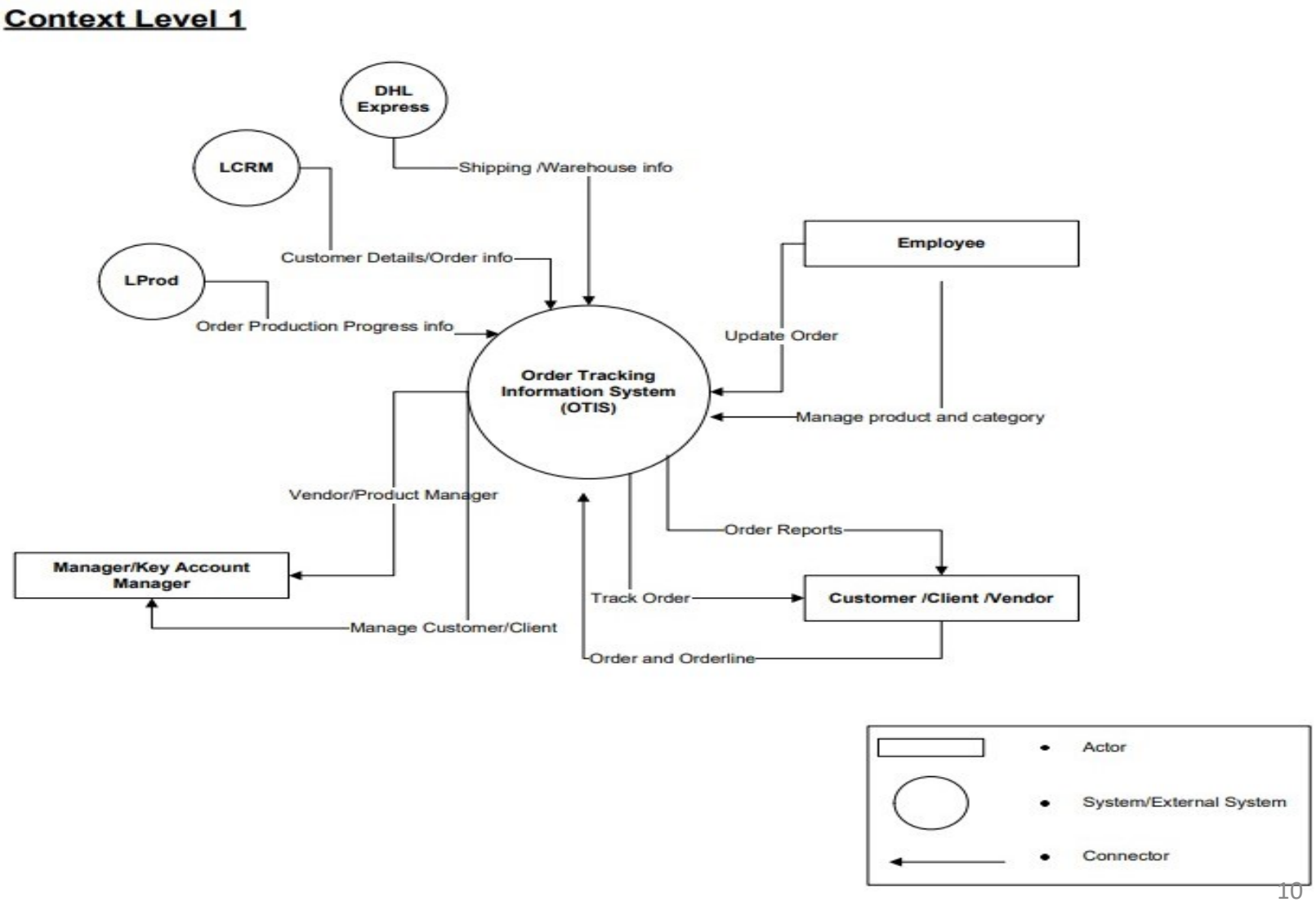
One of the main business goal of the user has a transparency in their order so they can view their order stages as soon as possible. Our system would be up and running in working hours 8:00 – 18:00 to provide user to check their order status

Why?	Regular data sync required between OTIS,LCRM,LPROD
Source of stimulus	System Developer/ System Administrator
Stimulus	New module integration and upgrading new functionality
Environment	Production/Go Live
Artifact	OTIS, LPROD,LCRM
Response	Ability to make necessary changes
Responsemeasure	System should be available within 20 minutes (Downtime)

Use Case Diagram



Context Diagram



Architecture concerns and other drivers :General Concern

• OTIS system has data dependency on DHL Express system. Our system could fail if there is any change in system.

• Our system will keep logs of the event and save them in server in logging directory. System Admin would have the access to view the logs through interactive UI.

• Small modular structure will give us easy to check which component/module isn't working by using exception handling in the system.

• Our system would down at midnight time 12:00 AM-12:20 AM. IT team working at this team is needed.

• We have a layered architecture. Frontend Team would depend upon the backend team. If change is needed, frontend wait for the backend team to implement the change.

Constraints:

Technical Constraints / Runtime Interface Requirements

Following applied technologies - operating systems - middleware - databases - programming languages:

SR No	Constrains	Justification
C1	Windows/Linux	As our system would be web based, web browser would need an User interactive Operating system for displaying the data
C2	Programming Language: Java Spring Boot	Our existing system is made in Java Spring Boot and our In-house IT team has an expertise in Web development. So developing OTIS would be result in quality product.
C3	SQL Server	SQL server would be used in getting and retrieving the data of order and users
C4	DHL express API to get the customer order tracking and warehousing details	Order tracking data would come from external system as we do not manage the data. We have to rely on their given API endpoints
C5	LPROD module API to get the production details of customer order	The production data of orders would come from external system and will have to rely on their data for our system to work
C6	6 months for develop and integrate the system	As the time given by the Lysia owner to develop the system within 6 months, we need to develop and integrate the system

Why the selected pattern is appropriate for the problem

As our prioritized quality attributes are **performance**, **maintainability** and **availability**, our solution would be **Layered based system**. The layered based architecture system is easy to learn as the time of the developing, testing and deployment time is 6 months. Our developers are already familiar to develop this architecture. This reduces the dependency on the modules/components on each other which increases the cohesion in the system. Cost overhead, testing is low and error handling is relatively easy.

Design Decisions and Rationale

Design Decisions	Rationale
OTIS system is hosted in a production server and multiple users that are use portal by various departments.	Due to network latency, the overhead brought on by intermediaries who handle communication, anoverall architecture approach in OTIS may have a detrimental effect on the performance of an application. To ensure that the required performance criteria are met, both the user as well as the component provider should indeed carefully develop and assess the architecture.
Divide domain objects into generic and specific components.	Complete functional sets are represented by application components, but these functional sets are maintained by smaller elements that are found inside the layers. In this pattern, the "components" were also what have been referring to it as components. Module specialization is related to the layers in which they are found (e.g., UI modules). There are no great options to breaking down the layers into functional modules.

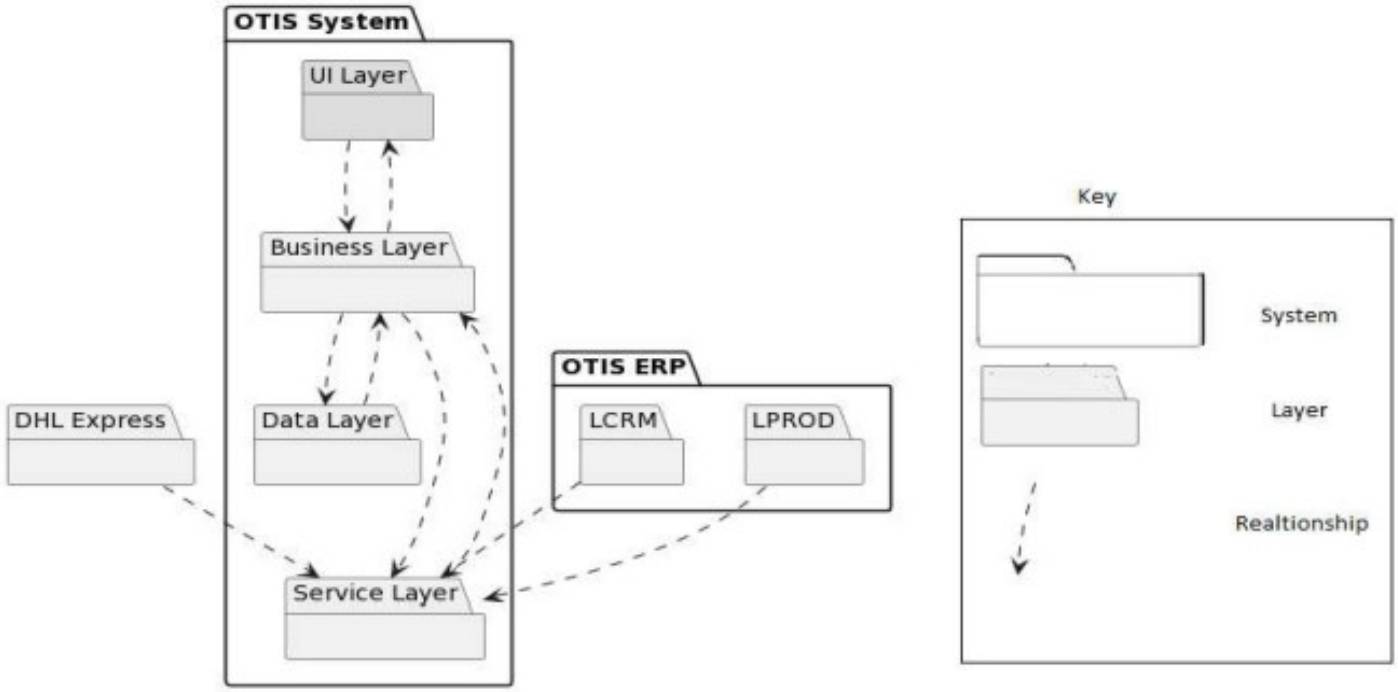
Design Decisions and Rationale Continue…

Develop the system in web based (Java) as the developer team is already familiar to the programming paradigm. the development team will be implement better approach to enhance the performance of OTIS. Java spring boot is more capability to handle module performance via front end.	Performance

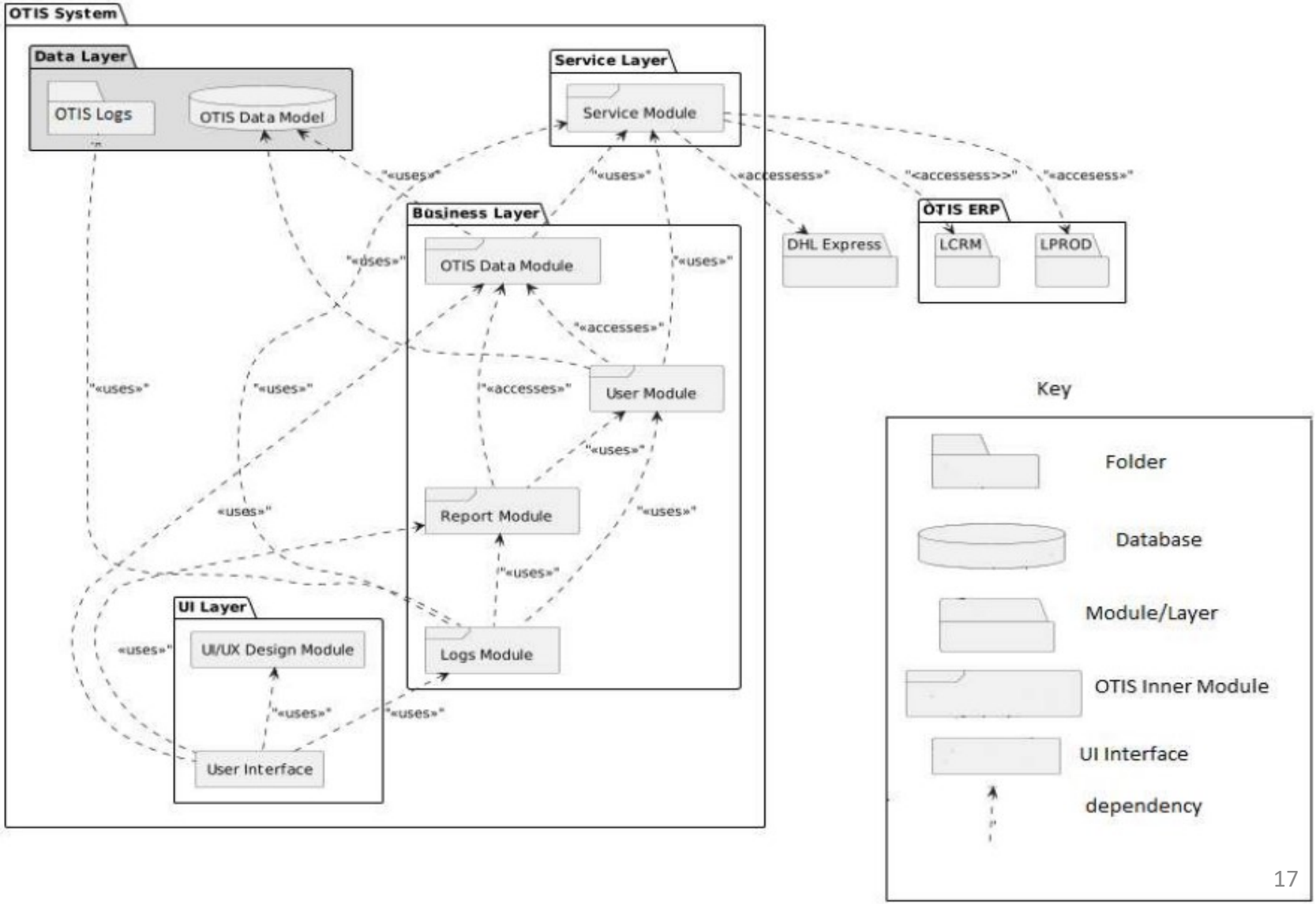
Module View: Primary Presentation

We have selected uses view from the model view because our architecture is layer-based architecture and it will be better to select this architecture because we want to see the interaction of different modules and services and how these layered are interacted.

- **UML Modular View Diagram**



Module View: Uses view



Module View: Rationale/Architectural Drivers

Architectural Drivers:

Design Decision	Addresses	Rationale
Layered-oriented architecture: Separate layers in the system	QA2: Maintainability	As our main three major quality attributes are Performance, Maintainability and Availability. Given the conditions it will be easy to develop and maintain. As dependency is low, it addresses availability. Performance will be good as system would have less than 10000 users.
Using Web Interface	Concerns C5	Delivery time is 6 months. Our team is trained to build Web application
Create Separate Report Module in the system	UC2: Generate Order Report	Report record and report UI is created separately from the system. It increases cohesion and it would be easy to keep

Module View: Tactics

Tactics: (Change tactics with the performance)

We as a team looked into the quality attributes, use cases, constraints and concerns implemented

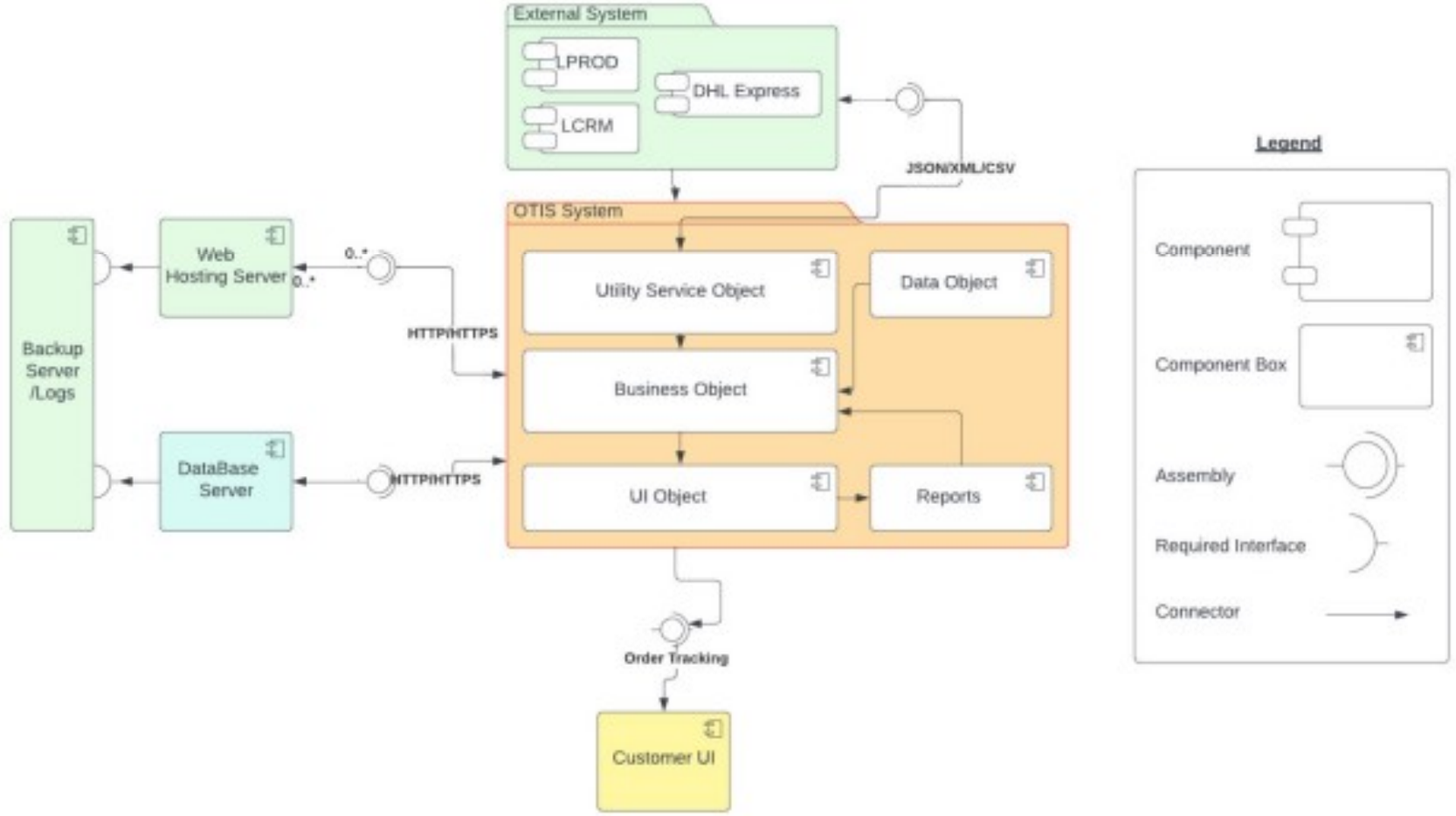
Tactics	Tactics QA	Module Implemented
Create a database backup and update it weekly	Performance(Maintain multiple copies of Data)	Data Model(Data Layer)
Trigger data dump of users from LCRM once every 4 weeks	Performance(Manage Event Rate)	Service Module
Expectation handling is done if an error occurs in the generation of reports and alert is notified to System Admin	Maintainability(Fault Detection)	Report Module
If the database doesn't have the required order tracking data, an API call is generated to the external system LPROD,DHL	Availability (State Resynchronization)	OTIS Module
System Administrator can view the error logs along with the timestamps and module	Availability (Expectation Detection)	Logs Module
Service module and Database server will be monitored and how well the system is performing	Availability (Conditional Monitoring)	Service Module, Database Server
Modules is split to increase the cohesion in the system	Maintainability (Split Module)	Business Layer

Other Architectural Decisions:

Decisions	Rationale
UI/UX module is used by the user interface in importing UI components	As there are multiple stakeholders in the system, there are multiple dashboard for the system. We use pre-build UI/UX components.

Component & Connector : UML C&C Diagram

- **UML Component and Connector Diagram**



Component & Connector : Rationale

Architectural Drivers:

Design Decision and location	Addresses	Rationale
Select separate service component for external system	Quality Attribute (Maintainability) QA1	The system shall cover maintainability quality attributes so separate external component is easy to build and deploy as service. If any other new service need to attached the only process will be added as factory pattern implemented so it's easy to maintain and component efficiency is very high so not dependence on main OTIS system. The background service constantly run and data extract and upload to system with 10 min time span.
Select Responsive web Interface on OTIS System UI Component	Concerns	The system shall build with responsive design so it's easy to use and response on any web browser and mobile devices. The quality of web responsive interface is light weight and scalable.
Select separate Reports Component	UC2	The system shall build with report component separately because development and maintenance is easy and other 3rd party libraries like DevExpress and Telerik tools charts/ pivot grid easily integrated with the reports component

Component & Connector : Tactics

In designing for OTIS system components and connectors we deeply look into QA, use cases, constraining and implementation approaches

Tactics	Tactics QA	Module Implemented
Log management on event of success and failure.	Availability	OTIS Business Object Component
UI assign for managing user and assign screens to customers	Performance	UI Component
Check records to dump data to db server on every 10 min	Performance	Utility Service Component
Local and global exception handling	Maintainability(Fault Detection)	Business Object Component
External API Expose for HTTP/HTTPS web services	Availability	External Service component

Other Architectural Decisions:

Decisions	Rationale
Customer UI via UI Object Component	As we build UI Object component for customer so the admin would be assign the custom UI and roles and rights for customer for view reports and tracking the order.

Questions & Answer


Thank you