Gusher Oil

1 Background

The Oil and Gas industry is under constant pressure to innovate – geopolitical instability, environmental concerns, and rising energy prices and consumption have created a business environment where inflexible oil companies are punished by a combination of free market pricing and government sanctions, but where nimble and flexible companies can reap monetary rewards. Gusher Oil is an oil company headquartered in the United States, with important operations in Russia, Africa, and the Gulf of Mexico. The company has recently entered into an agreement with a large oil auctioning marketplace to bring its product to market more efficiently, improving cash flow and reducing the cost of distribution. The internal business case at Gusher Oil projects cost savings of 20 percent over five years and improved margin and yield management of 7 percent per annum over the current steady state of the business, making this one of the most important projects currently under development at the company.

2 Workshop Output

You are the architect for the marketplace integration project at Gusher Oil. You have been tasked by Gusher Oil management to lead the team responsible for the design, implementation, and ongoing management of the complete system as a turnkey or complete solution.

After an intensive series of discovery workshops with the in-house business analysts and subject matter experts, you know the following facts:

- The oil auctioning marketplace exposes a Java™ technology-based API using Java Message Service (JMS) technology to allow companies to send messages placing oil for sale and to bid for oil with characteristics, such as guaranteed messaging, message acknowledgment, and message security available.
- Gusher Oil already runs a complex pricing system in-house that calculates what the price for each placement of oil in the auction marketplace should be (based on how it was extracted and its distance and likely transportation method to its final destination point). As the solution architect, you need to integrate with this system using web services to price the oil before making it available to the auction market place for sale.
- The final remaining external system is Gusher Oil's inventory management system that is also accessed using web services and will allow you to see what unsold capacity remains available for auction.
- The actual placement of orders for the auction management system are handled manually – Gusher Oil employs a team of traders who track the rise and fall of oil prices and use a combination of timing and pricing information and other systems outside the scope of this project to determine the best time to buy or sell oil placements.
- System performance (99 percent of all messages to be constructed and sent in three seconds or less to the IP address of the API server), scalability (to 200 concurrent users), availability (99.999 percent during core working hours) and security (128-bit encryption at a minimum) are all key requirements and you must explicitly address each requirement in your proposed solution.

2.1 Business Domain Model

The following business domain model describes the key objects identified during the workshops. All of these objects and the relationships between them should be addressed in your design and implementation.

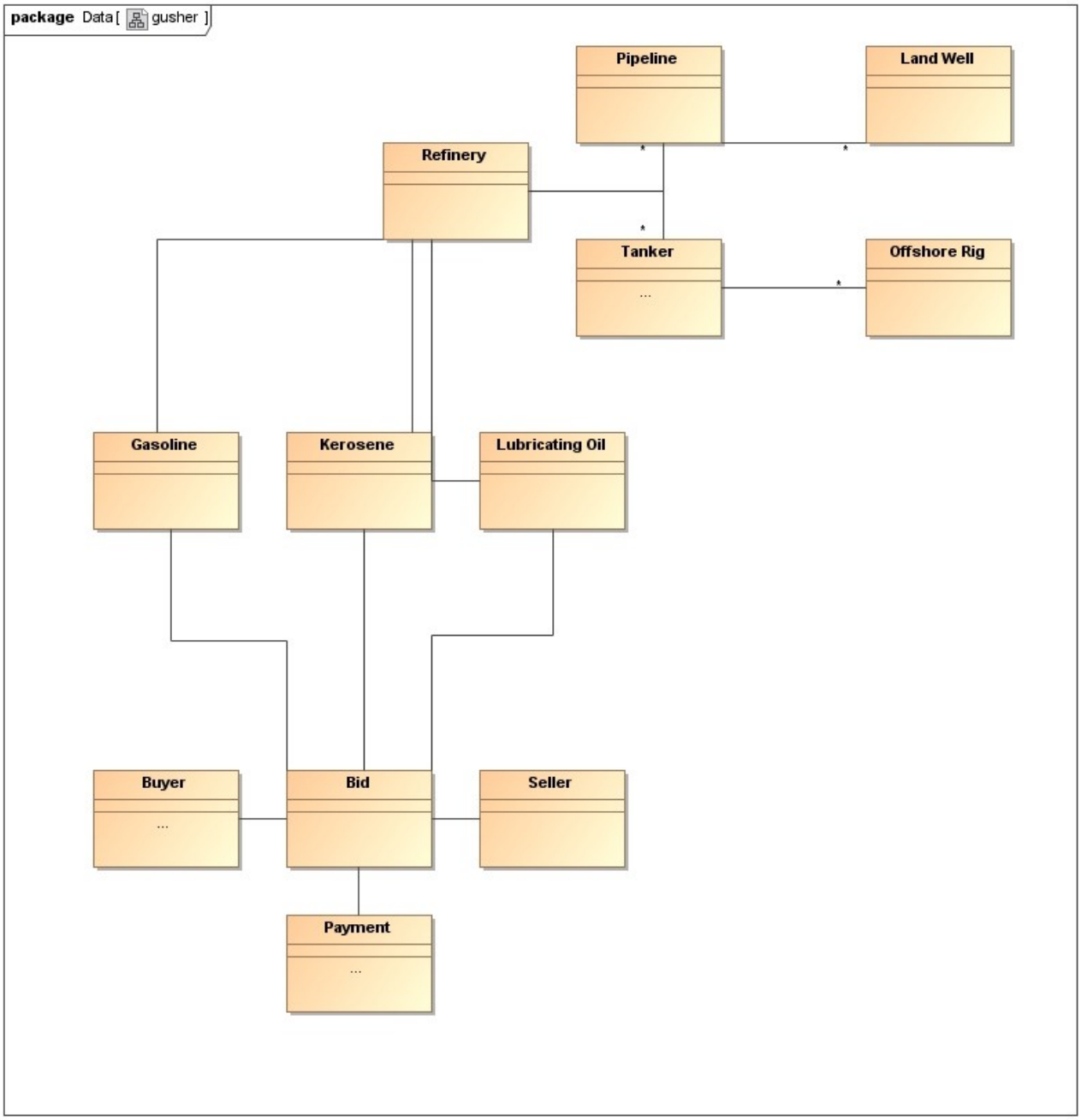

Illustration 1: Gusher Oil Auction Marketplace Project Business Domain Model

2.2 Use Case Diagrams

The following use case diagram captures the main use cases that must be supported by your proposed design and implementation.

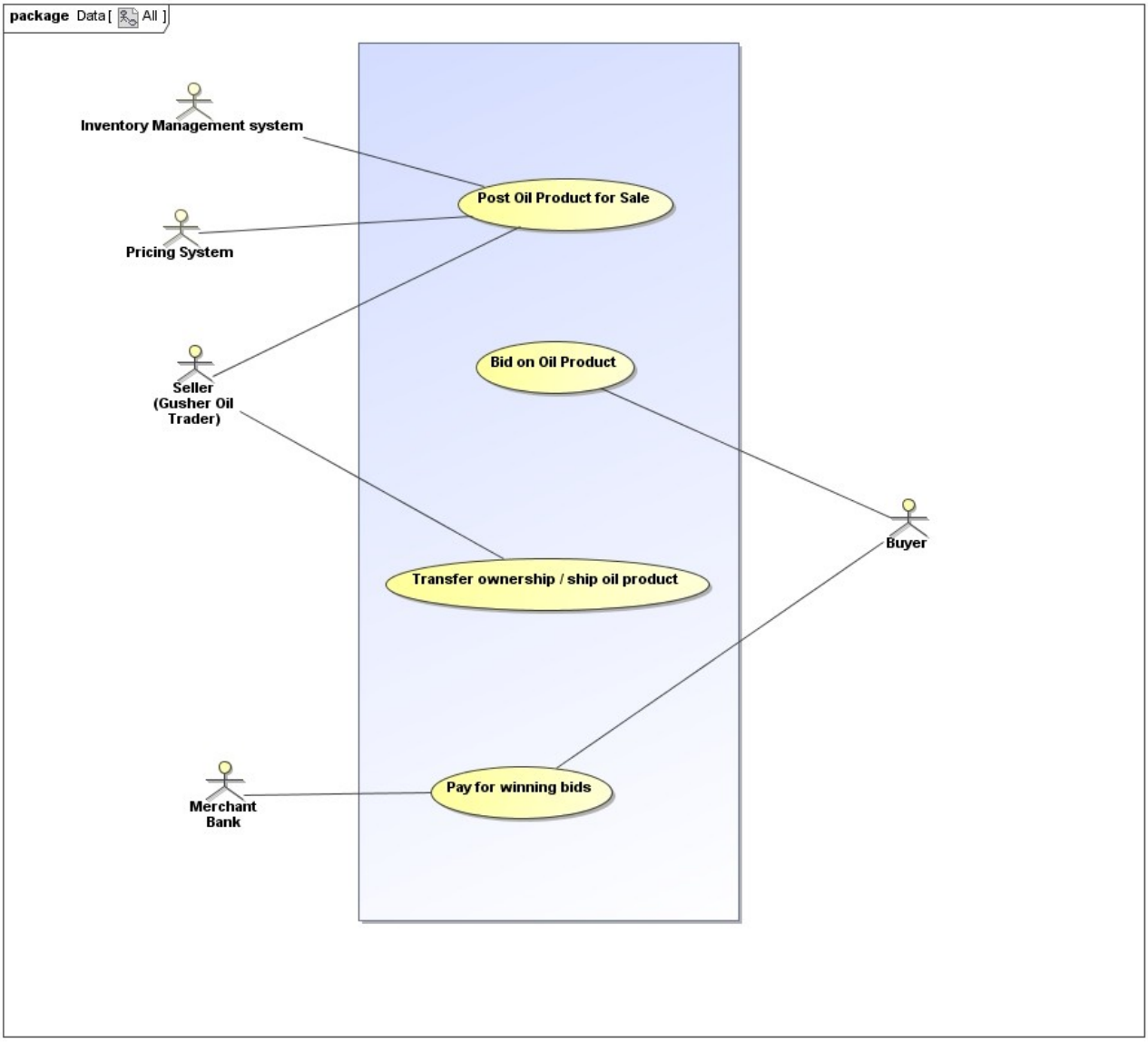

Illustration 2: Gusher Oil Auction Marketplace Use Case Diagrams

The use case specifications providedare a first pass at the use cases and provide enough detail in order for you to architect a solution.

Use Case Specification – Post Oil Product for Sale

Brief Description

The Post Oil Product for Sale use case allows an authorized Gusher Oil representative to place a quantity of oil product for sale in the marketplace.

Basic Flow

1. A Gusher Oil representative inputs a quantity of oil product.
2. System responds with a validation that the quantity and product type represents a valid item for sale in the marketplace.
3. The system makes the sale offer open so that potential buyers can view and bid for the oil product.

Use Case Specification – Bid on Oil Product

Brief Description

The Post Request use case allows authorized potential buyers to place bids for a specific quantity of oil product.

Basic Flow

1. The potential buyer searches for offers matching some desired criteria (for example, quantity, price, quality, location).
2. System responds with all matching offers.
3. The potential buyer selects a specific offer of interest and places a bid for the product.
4. The system records the expression of interest against the offer.

Use Case Specification – Transfer Ownership / ship oil product

Brief Description

The Transfer Ownership / ship oil product use case allows an authorized Gusher Oil representative to transfer ownership of a purchased offer directly to the purchaser or to a named intermediary for transport to a destination.

Basic Flow

1. A Gusher Oil representative selects a specific closed, successful offer.
2. System responds with the selected offer details.
3. The Gusher Oil representative changes the ownership status of the oil product under consideration from Gusher Oil to the buyer / the buyer's transport company.
4. The system records the status change and unlocks the oil product so that it can be accessed by the buyer / the buyer's transport company.

Use Case Specification – Pay for Winning Bid

Brief Description

The Post Request use case allows an authorized Buyer to pay for an oil product won at auction.

Basic Flow

1. The successful buyer executes a request for the successful bid.
2. System responds with the successful bid details.
3. The successful buyer enters payment details against the bid.
4. The system records the details and forwards the information to the merchant bank, initiating a transfer of funds between the successful buyer and Gusher Oil.

3 Deliverables

It is your task to create an architecture and design for the System under Development (SuD) with the given business domain model, information provided above and requirements in the use cases. The architecture must be based on the Java™ Platform, Enterprise Edition (Java EE) platform[1]. All deliverables will be accepted as HTML only and each diagram must be UML compliant.

1. Create a class diagram for the SuD. Public method names referenced in other UML diagrams (e.g. sequence diagrams) should be provided.
2. Create a Component diagram for the SuD showing the components used in the system and their interaction. Examples of components are Enterprise JavaBean™ (EJB™), servlets, JavaServer™ Pages (JSP™), major Plain Old Java Objects (POJOs), and important Managers / Controllers / Design Pattern implementations.
3. Create a Deployment diagram that describes the proposed physical layout of the major tiers of the SuD.
4. Create either a Sequence or Collaboration diagram for each use case provided.
5. List the top three technical risks you have identified in the project and identify a mitigation strategy for each risk.
6. List any assumptions made during the process of coming up with the architecture and design that have a material impact on your design and any decisions made in arriving at that design.

Your architecture and design will be graded on how well it supports the requirements detailed in this document and on the clarity of all information provided in both textual and diagrammatic form.

3.1 Submitting Your Work

Note to the candidate: Failure to follow the rules described here will result in an immediate failure and require a re-submission on your part.

When you have completed your solution, you should have an "index.html" that has your name, ID, a link to the class, component, and deployment diagrams, risk list, and a link to each of the sequence/collaboration diagrams. Build a JAR archive that contains all html files. You must build a JAR archive; do not send individual files.

The name of your submission JAR archive file MUST be derived from your testing ID. Your archive file name MUST BE scea-AAAAAAAAA.jar, where AAAAAAAAA is your testing ID.

4 Marking

The tentative marking for this beta is described below. The markings will be finalized based on the results of the beta. You will receive a single score for parts 2 and 3.

Below are the sections of the assignment and points available per section with a minimum score per section if it is a required section to pass. Minimum score to pass the exam, if passing required sections, is 114 (tentatively set). If a person fails any of the required sections Component, Class, Deployment or Question, then it is an automatic failure regardless of the final score. We will only post a pass or fail. If you fail, we will let you know which sections need improvement in order to pass so you do not have to focus on all sections with your re-submission.

Points Minimum Pass

1 While we do not explicitly require Java EE 5, we expect that J2EE™ 1.4 or higher to be used.

Component Diagram	40	26
Class Diagram	40	26
Deployment Diagram	24	17
Interaction Diagrams	16	0
Risk & Mitigation List	16	0
Part 3 Short Answer	24	17
Total	160	