Task – Communication in warehouse system

Functional requirements

A generic warehouse management system provides logistics support to manage the flow of items and assets in, and across, warehouse storage facilities. The system is used by many different customers and applications at the same time to store their products in the warehouse. The warehouse management system is responsible on stock management, order management, receiving and shipping, and the management of storage and transportation facilities in the warehouse, as well as operational tasks such as the execution of concrete transportation orders. In addition, the system generates data for reporting to management.

The applications, which use the warehouse system are different in their requirements and interests. For example, some applications submit ordering requests for products. In this case, the system needs to process the order, retrieve the products from the warehouse, and ship the products to the customer. On the other hand, other applications are only interested to list the products in stock.

The software architect created the first base-line architecture based on the functional requirements. The design consists of 3 layers:

- *Applications layer*: It contains applications used by companies to access the warehouse management services.
- *Business process layer*: It contains the different business processes and logic provided by the warehouse management.
- *Persistence layer*: It's responsible on storing the different data, which will be used for reporting and logging.

Non-functional requirements

Interoperability: The entire IT infrastructure is inherently distributed. The architectures of all software systems must therefore take into account that their partner systems are remote and only accessible via some form of Inter-Process Communication (IPC)..

Extensibility: The system should be designed to allow developing new applications at the customer side. New applications could have different interests for services provided by the warehouse management. Integrating a new application with the system should be dynamically configured and developed, and not take more than 2 weeks of development.

Scalability: Warehouses can differ significantly in their size.

- The warehouse management system must therefore be able to support small warehouses with just a few thousand users, as well as large warehouses with well over a million users. Warehouses can also vary in the functionality they need. For example, depending on the applications developed by customers, a warehouse management process control system must provide more or less powerful operational functionality.
- The logging mechanisms and data retrieval need to be scalable for high throughput.

Performance: The system must ensure that all transportation orders are executed in a timely and efficient manner without any visible interruption or stop-and-go behaviour.

Availability. The warehouse operates in 24/7 mode with three shifts per day. Availability is therefore crucial for supporting the business case for a warehouse management process control system. Any downtime disrupts supply chains, the state and operation of other systems, people, and so on, which ultimately means loss of business and money. Industrial automation systems in general, and process control systems specifically, therefore, typically demand a minimum availability of 99.999%—a maximum downtime of just over five minutes per year!

Constraints:

- Database servers cannot be upgraded with new memory or processors.
- The different software systems in an industrial automation environment are often provided by different vendors, however, which in turn requires appropriate application integration measures to support end-to-end operations seamlessly across multiple systems.

Task goal

The architect would like to design and decompose the logical and physical architecture of the system. The logical architecture specifies the components of the system, and their relationships. While the physical architecture specifies the system components and their deployment on servers. Propose and model the logical and physical architecture for the system, which fulfils the requirements and constraints. In your design, specify and justify the components, their relationships, and the different architectural solutions (e.g. patterns), which you used in your design.

To design the logical architecture of the system, follow the following steps:

1) Identify possible components of the system. A component could be an application, a user interface component, a database or a processing component. Align your components with the given layer structure.

2) **Use the following patterns to determine the communication mechanism between components/layers**:
   a. Publish-subscribe architecture pattern.
   b. Queue architecture pattern.
   c. Synchronous mechanisms (RPC and REST).

To design the physical architecture of the system, **use the following solutions**:

   a. Component replication performance tactic.
   b. Active redundancy availability tactic.
   c. Tiered distribution architecture pattern. (https://msdn.microsoft.com/en-us/library/ff647195.aspx).

The solution of the task is expected to be one or two architecture models and an explanation for the architecture, which justify the design decisions.