A quick reference material

Kallil de Araujo

June 15, 2023

Contents

Abstract

When studying for job interviews I frequently had to spend quite some time researching possible concepts that could come up. So, to make my life less complicated, I created this file with the objective of helping me save some time when studying for those situations.

Chapter 1

Methodologies

In the IT context, a methodology refers to a structured approach or set of principles and practices that guide the planning, development, implementation, and management of IT projects or processes. It provides a systematic framework for organizing and executing tasks, as well as a common language and set of tools to facilitate collaboration among team members. A methodology encompasses a range of activities, techniques, and guidelines tailored to address specific challenges and ensure successful outcomes in IT projects.

A methodology typically outlines a step-by-step process for undertaking IT initiatives, including project initiation, requirements gathering, design, development, testing, deployment, and maintenance. It may incorporate various methodologies or frameworks, such as Agile, Waterfall, Scrum, or Lean, depending on the specific needs of the project. The choice of methodology often depends on factors such as project size, complexity, team structure, and organizational culture.

A well-defined methodology brings several benefits to the IT context. It promotes consistency and standardization in project execution, allowing for more predictable outcomes and better resource management. It helps identify and mitigate risks early in the project lifecycle, facilitating proactive decision-making. Additionally, a methodology supports effective communication and collaboration among project stakeholders, ensuring shared understanding and alignment of objectives. By providing a structured approach, a methodology enhances project transparency, facilitates progress tracking, and enables continuous improvement through feedback and lessons learned.

Add something about SIX SIGMA

1.1 Agile

Source!

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
That is, while there is value in the items on the right, we value the items on the left more.

Agile is an iterative and collaborative approach that focuses on flexibility and adaptability. It emphasizes frequent communication, incremental development, and delivering working software in short iterations.

1.1.1 Scrum

Scrum is an Agile framework that utilizes cross-functional teams to deliver software incrementally. It involves iterative sprints, daily stand-up meetings, and continuous collaboration to foster flexibility and rapid development.

1.1.2 Kanban

Kanban is a visual project management approach that focuses on continuous delivery and workflow optimization. It uses a Kanban board to visualize tasks, limit work in progress, and optimize the flow of work.

1.2 Waterfall - measure twice, cut once

Source!

Waterfall is a sequential and linear approach where each phase of a project is completed before moving to the next one. It follows a structured process with defined milestones, making it suitable for projects with well-defined requirements and minimal changes.

In other words, the waterfall methodology is a project management approach that emphasizes a linear progression from beginning to end of a project. This methodology, often used by engineers, is front-loaded to rely on careful planning, detailed documentation, and consecutive execution.

The Waterfall methodology — also known as the Waterfall model — is a sequential development process that flows like a waterfall through all phases of a project (analysis, design, development, and testing, for example), with each phase completely wrapping up before the next phase begins.

It is said that the Waterfall methodology follows the adage to "measure twice, cut once." The success of the Waterfall method depends on the amount

and quality of the work done on the front end, documenting everything in advance, including the user interface, user stories, and all the features' variations and outcomes.

1.2.1 5 common stages in a Waterfall process

The Waterfall methodology follows a chronological process and works based on fixed dates, requirements, and outcomes. With this method, the individual execution teams aren't required to be in constant communication and, unless specific integrations are required, are usually self-contained.

Team members also tend to work independently and aren't expected to provide status reports as often as with the Agile approach. Usually, one phase doesn't begin until the previous one is finished.

Requirements

Design

Implementation

Verification or Testing

Deployment & Maintenance

1.3 Lean

Lean methodology aims to eliminate waste and maximize value in the software development process. It emphasizes continuous improvement, customer focus, and efficient resource utilization.

1.4 DevOps

DevOps combines software development (Dev) and IT operations (Ops) to enhance collaboration and streamline the software delivery lifecycle. It focuses on automation, continuous integration, continuous delivery, and close collaboration between development and operations teams.

1.5 Rapid Application Development (RAD)

RAD is a methodology that emphasizes rapid prototyping and iterative development. It involves user feedback and frequent iterations to quickly build and refine software applications.

1.6 Extreme Programming (XP)

XP is an Agile methodology that emphasizes close collaboration, continuous feedback, and a high degree of customer involvement. It emphasizes practices like test-driven development, pair programming, and frequent releases.

1.7 Spiral

The Spiral methodology is a risk-driven approach that combines elements of both waterfall and iterative development. It involves iterative cycles where the project progresses through planning, risk analysis, prototyping, and customer evaluation.

1.8 Feature-Driven Development (FDD)

FDD is an iterative and incremental methodology focused on delivering tangible features quickly. It emphasizes domain modeling, iterative development, and feature-level planning and tracking.

Chapter 2

Important ETL Concepts

In the context of IT, ETL stands for Extract, Transform, Load, which is a process used to integrate and consolidate data from various sources into a target system or data warehouse. ETL plays a crucial role in data management and enables organizations to extract valuable insights and make informed business decisions.

The first step in the ETL process is extraction, where data is collected from different sources such as databases, files, APIs, or external systems. This involves retrieving relevant data based on defined criteria and requirements. Once the data is extracted, it undergoes transformation, which involves cleaning, validating, and structuring the data to ensure consistency, accuracy, and compatibility with the target system. Transformations may include data cleansing, data enrichment, data aggregation, and applying business rules or calculations.

Finally, the transformed data is loaded into the target system, typically a data warehouse or a data mart. This step involves mapping the transformed data to the appropriate tables or entities in the target system, ensuring data integrity and maintaining the overall data structure. The loaded data can then be used for reporting, analysis, and decision-making purposes.

2.1 Data Extraction

Understanding various methods and techniques for extracting data from different sources such as databases, files, APIs, or web scraping.

2.2 Data Transformation

Familiarity with data transformation techniques, including data cleansing, data validation, data enrichment, and data aggregation.

2.3 Data Quality

Knowledge of data quality assessment and improvement techniques to ensure accurate and reliable data in the ETL process.

2.4 Data Integration

Understanding how to integrate data from multiple sources into a unified and coherent format, resolving data inconsistencies, and handling data mapping and data merging.

2.5 Data Modeling

Proficiency in data modeling concepts and techniques, including dimensional modeling, relational modeling, and schema design to support efficient data storage and retrieval.

2.6 ETL Architecture

Understanding the overall architecture and components of ETL systems, including staging areas, data warehouses, data marts, and data pipelines.

2.7 ETL Tools and Technologies

Familiarity with popular ETL tools such as Informatica PowerCenter, Talend, SSIS, and understanding their functionalities, capabilities, and best practices.

2.8 Data Governance

Knowledge of data governance principles, policies, and practices, including data lineage, data security, data privacy, and compliance regulations.

2.9 Performance Optimization

Skills in optimizing ETL processes for performance and scalability, including data partitioning, indexing, parallel processing, and query optimization.

2.10 Error Handling and Logging

Understanding techniques for handling errors and exceptions during the ETL process, implementing error logging, and implementing appropriate error handling strategies.

2.11 Change Data Capture (CDC)

Knowledge of CDC techniques to capture and process incremental data changes efficiently, ensuring synchronization between source and target systems.

2.12 Metadata Management

Understanding the importance of metadata in ETL processes, including metadata extraction, storage, and management for data lineage, data profiling, and data documentation.

2.13 Workflow Orchestration

Familiarity with workflow management tools such as Apache Airflow, Oozie, or Luigi for orchestrating complex ETL workflows and scheduling jobs.

2.14 Data Security

Knowledge of data security principles and practices, including encryption, access controls, and data anonymization techniques to protect sensitive data during ETL operations.

2.15 Data Warehousing

Understanding the fundamentals of data warehousing, including star schemas, snowflake schemas, slowly changing dimensions (SCDs), and dimensional hierarchies.

2.15.1 Slowly Changing Dimensions (SCDs)

Slow Changing Dimensions refer to the handling and management of slowly changing data in a data warehousing environment. In ETL processes, SCDs are used to track and manage changes to dimensional attributes over time.

SCDs are essential because they enable the capture and representation of historical data in a data warehouse. This is particularly useful when analyzing trends, performing historical comparisons, or conducting time-series analysis.

There are different types of Slow Changing Dimensions, including:

- Type 1 SCD: Overwrite the existing dimension attribute with the new value, effectively losing the historical information. This approach is suitable when historical data is not needed or when the dimension attribute is not expected to change over time.

- Type 2 SCD: Maintain a separate row for each change in the dimension attribute, creating a new record with an updated attribute value and an assigned surrogate key. This approach allows for historical tracking but can lead to data redundancy.
- Type 3 SCD: Add new columns to the dimension table to track specific changes, typically storing the current value and previous value of the attribute. This approach offers limited historical tracking but can help maintain simplicity in the data model.
- Type 4 SCD: Create a separate history table to store all changes to the dimension attribute, while the main dimension table only contains the current value. This approach provides comprehensive historical tracking while minimizing redundancy in the main dimension table.

Chapter 3

Some questions and answers

3.1 Data visualization

1. **Q**: What is the importance of data visualization in data analysis?
   A: Data visualization is crucial in data analysis as it allows us to present complex datasets in a visual format that is easily understandable. It helps to uncover patterns, trends, and outliers, enabling stakeholders to make informed decisions based on the insights derived from the data.

2. **Q**: How would you approach analyzing a complex dataset using Tableau?
   A: When analyzing a complex dataset using Tableau, I would start by understanding the dataset's structure and the specific questions or objectives at hand. Then, I would identify relevant variables, perform data cleaning and transformation if necessary, and create visualizations such as charts, graphs, or dashboards to explore the data and extract meaningful insights.

3. **Q**: Can you explain the process of developing interactive dashboards in Tableau?
   A: Developing interactive dashboards in Tableau involves several steps. First, I would identify the key metrics and KPIs to be included in the dashboard. Then, I would design the layout, selecting appropriate visualizations and arranging them in a logical and intuitive manner. Next, I would add interactivity through filters, parameters, and actions to allow users to explore the data dynamically. Finally, I would refine the dashboard's appearance and ensure its functionality across different devices and screen sizes.

4. **Q**: How do you ensure that your visualizations effectively communicate performance metrics and KPIs to stakeholders?
   A: To ensure effective communication of performance metrics and KPIs, I focus on simplifying complex information and presenting it in a clear and concise manner. I use appropriate visual encodings, such as color, size,

and position, to highlight key insights. I also provide context and relevant annotations to help stakeholders understand the data. Regular feedback from stakeholders is valuable in refining the visualizations and ensuring they meet their needs.

5. **Q**: What techniques do you employ to identify trends, patterns, and anomalies in data?
   A: To identify trends, patterns, and anomalies in data, I use various techniques such as data aggregation, filtering, and sorting. I also employ statistical analysis methods, including regression, correlation analysis, and hypothesis testing. Visual techniques, such as time series analysis, heat maps, and scatter plots, can be helpful in spotting trends and patterns. Additionally, I leverage Tableau's built-in features like trend lines, reference lines, and clustering to uncover insights in the data.

6. **Q**: How do you ensure data accuracy and reliability when analyzing and visualizing datasets?
   A: Ensuring data accuracy and reliability is essential in data analysis and visualization. To achieve this, I perform data validation and cleansing processes, checking for missing values, outliers, and inconsistencies. I also conduct data integrity checks, verifying the consistency of data across different sources or time periods. It's crucial to document the data sources, assumptions, and any data transformations performed to maintain transparency and enable reproducibility.

7. **Q**: Can you explain the concept of key performance indicators (KPIs) and their role in data analysis? **A**: Key performance indicators (KPIs) are quantifiable metrics that measure an organization's performance against its strategic goals. They provide insights into the success or effectiveness of specific processes, activities, or objectives. In data analysis, KPIs act as benchmarks and guide decision-making. By analyzing and visualizing KPIs, we can monitor performance, identify areas of improvement, and track progress over time.

8. **Q**: How would you handle a situation where a stakeholder requests a specific visualization that may not effectively represent the underlying data?
   A: In such a situation, I would begin by understanding the stakeholder's requirements and the reason behind their request. Then, I would communicate and collaborate with the stakeholder, explaining the limitations or potential issues with their requested visualization. I would offer alternative visualizations that better represent the data and provide meaningful insights. It's important to have open and constructive discussions to find a solution that meets both the stakeholder's needs and the data's integrity.

9. **Q**: How do you stay updated with the latest trends and best practices in data analysis and visualization?
   A: To stay updated with the latest trends and best practices, I actively

engage in professional communities, attend industry conferences, and participate in webinars and workshops. I follow thought leaders and experts in the field, read relevant books and articles, and explore online resources, such as blogs and forums. Additionally, I continuously explore and experiment with new tools and techniques to expand my skill set.

10. **Q**: Can you provide an example of a challenging data analysis and visualization project you have worked on in the past and how you overcame the challenges?
    A: [Candidate's personal example or hypothetical scenario]

Chapter 4

SQL QUESTIONS

This chapter is focused on explaining some more advanced concepts, so assuming that you are already familiar with SQL basic concepts we will move forward and present some questions that I faced during job interviews and some others that I found on the internet.

4.1 Views vs Stored Procedrures

In SQL, views and stored procedures are both database objects that serve different purposes. Here are the key differences between views and stored procedures:

Purpose:

Views: A view is a virtual table that is derived from one or more tables or other views. It provides a way to present data from underlying tables in a customized or simplified manner. Views are primarily used for querying and reporting purposes.

Stored Procedures: A stored procedure is a named set of SQL statements that are stored in the database and executed as a unit. It is used to encapsulate a series of database operations and logic that can be called and executed repeatedly. Data Modification:

Views: By default, views are typically used for reading data rather than modifying it. While you can define an "updatable view" that allows certain types of modifications, such as updating or deleting rows, the level of flexibility is limited compared to stored procedures.

Stored Procedures: Stored procedures can include data modification statements (e.g., INSERT, UPDATE, DELETE) to modify the data in tables. They allow for dynamic and complex data manipulation within the procedure. Execution and Result:

Views: Views are treated as table-like objects and can be used in queries like a table. When a query references a view, the underlying query associated with the view is executed, and the result is returned as if querying a table.

Stored Procedures: Stored procedures are explicitly executed by calling their

name using the EXEC or CALL statement. They can return multiple result sets or output parameters. Persistence:

Views: Views do not store any data themselves. They are defined queries, and their results are dynamically generated based on the underlying tables or views each time they are referenced. Views do not persist data. Stored Procedures: Stored procedures are saved in the database and persist even after the database session ends. They can be called and executed whenever needed. Parameters:

Views: Views do not accept parameters directly. They are predefined queries that return a result set based on the defined query structure and underlying tables/views. Stored Procedures:

Stored procedures can accept input parameters, allowing you to pass values into the procedure when it is called. Parameters provide flexibility to execute the procedure with different input values.

In summary, views are primarily used for querying and reporting data in a customized manner, while stored procedures are used to encapsulate and execute complex operations and logic. Views provide a virtual representation of data, while stored procedures allow for dynamic data manipulation and can accept parameters.

4.2 Intermediate and advanced concepts, Q&A