**Answer key: Radhakrishnan S**

**State the 3 main software quality attributes? Justify.**

There are different attributes concerning the properties of critical systems and the best methods to develop them:

Performance —

Availability -

Dependability —

Security —

Modifiability —

The most appropriate definitions as much as applicable to the University case study.

**Define the primary tactics you will use to satisfy these quality attributes**

**Availability:**

Diagram

Description automatically generated

**Performance**:

Diagram

Description automatically generated

**Security:**

Diagram

Description automatically generated

Exact reproduction of this diagram is discouraged. The student should really understand the real meaning of what each tactic means will get full credit.

**Briefly explain the strategies to protect how intruders would be protected from exporting important files through mail or Google drive**

**Your content is stored securely**

When you upload a file of any type to Google Drive, it is stored securely in our world-class data centers. Data is encrypted in-transit and at-rest. If you choose to access these files offline, we store this info on your device.

Your Google Account comes with built-in security designed to detect and block threats like spam, phishing and malware. Your activity is stored using strong industry standards and practices.

### Drive uses data to improve your experience

To provide services like spam filtering, virus detection, malware protection and the ability to search for files within your individual account, we process your content.

To improve Drive’s performance and reliability, and to help with troubleshooting in case of issues while you use Drive, we collect performance data and crash analytics. We also save this info to help prevent abuse of our services and for analysis.

Anything which need not be an exact reproduction – even paraphrasing would get full credit.

**Identify two software quality attributes that can be traded off at the expense of other. Point exactly the logic with good reasoning**

**Software Quality Attribute Trade-offs**

Designers need to analyze trade-offs between multiple conflicting attributes to satisfy user requirements. The ultimate goal is the ability to quantitatively evaluate and trade off multiple quality attributes to arrive at a better overall system. We should not look for a single, universal metric, but rather for quantification of individual attributes and for trade-off between these different metrics, starting with a description of the software architecture.

The ensuing picture provides a good guideline for understanding the quality attributes trade-off process.

The answer should highlight how one parameter trades off the other parameter**.**

**Diagram

Description automatically generated**

**Define the term tactics as applied to dealing with quality attributes. Is there consideration of trade-offs while a tactic is being considered during design phase?**

A tactic is a design decision that influences the achievement of a quality attribute response—tactics directly affect the system’s response to some stimulus.

The focus of a tactic is on a single quality attribute response. Within a tactic, there is no consideration of trade-offs. Trade-offs must be explicitly considered and controlled by the designer. In this respect, tactics differ from architectural patterns, where trade-offs are built into the pattern.

Anything on these lines would do

**Define maximum three points that define a "Good Architecture"**

* Well defined modules whose functional responsibilities are allocated on the principles of
* Each module should have a well-defined interface
* Quality attributes should be achieved using well-known architecture tactics specific to each attribute
* If an architecture depends upon a commercial product, it should be structured such that changing to a different product is inexpensive.
* Creating / Consumption of data should be separated in different modules
* Well defined processes or tasks that do not necessarily mirror the module decomposition
* The architecture should feature a small number of simple interaction patterns

Any three on these lines would do