Architectural tactics or strategies - Notes

Keorapetse Shiko

09 March 2016

1 Architectural Tactics

1.1 Introduction

Quality requirements have a significant influence on the software architecture of a system. Architectural tactics are techniques that an architect may use to comprehensively achieve the quality requirements. For each quality requirement listed, potential tactics will be mentioned.

1.2 Quality requirements

- Performance
  - The system is fairly small and simple. Any form or action of enhancing performance may be relatively negligible but a couple of tactics can be implemented.
  - Tactics
    - Spread load across resources
      - Concurrency can be introduced. Processing different streams of events on different threads and creating additional threads to process different sets of activities can be done by processing requests in parallel. Appropriately allocating threads to resources can maximally exploit concurrency.
    - Resource re-use, caching.
      - Considering the Readers/Writers problem, multiple readers will have access to view the resource while only one writer can have access to edit the resource. Multiple copies of data can be maintained. Caching is a tactic that replicates data which would be on the same repository to reduce contention.
- Reliability
  - Tactics
    - Resource locking
      - A resource or a part/section of the resource will be locked, only one user will be able to edit that resource. The other users will be notified about the lock by getting conflicts when they try to pull the resource.
    - Maintain backup

            - The database will have a back up of all of the information that it stores.

- Scalability
    - Tactics
        - Increasing capacity of communication channels
            - Users will access the system through a website front-end on a web server maintained by the department. The site will be browser independent in order to increase communication channels. The Android system will also be used as a replacement for the website ans vice versa which allow use of mobile phones or tablets.

- Security
    - A hierarchical system will be used to determine the security privileges of users of the system.
    - Passwords are to be hashed using at least sha256 and should be stored as such within the database along with a salt.
    - An inactive user session should be terminated after a period of 10 minutes with no activity.
    - A user who has forgotten their passwords can use a password reset option which will send a one time password to their registered email address so that they may login once using it and reset their password.
    - Tactics
        - Limit access(Authentication, authorisation, Minimise access channels?)
        - Minimize complexity
        - Event logging and analyses
        - Drop connection
        - Restore state
        - Encryption(hashing passwords)

- Flexibility
    - The client has stated that the system is not needed to be able to extend to accommodate a greater number of departments.

  - Tactics
    * Contract based
    * Dependency injection

- Maintainability
  - The system should have as few bugs as possible so as to prevent having to constantly maintain it in the future.
  - The system should be built in a modular way so that all services are decoupled in such a manner that allows for the extension of the system at a later stage.
  - Tactics
    * Localise changes(Anticipate expected changes)
    * Prevention of ripple effect(Use intermediary, maintain existing interface)
    * Defer binding time(Runtime registration/lookups)
    * Polymorphism, adherence to defined protocols

- Auditability/monitorability
  - Every action performed by a user should be logged and all details about said action should be stored.
  - These actions should be visible to admin users.
  - Tactics
    * Logging

- Integrability
  - User's document entries should not be able to be deleted, if it is a case where the document will not be completed it should remain in the system and be terminated.
  - A user with no admin rights should not have access to admin privileges so that the system's data may remain integrable and safe.
  - The system should not ever be in a state where it is under pressure and the data is at risk of becoming corrupted. The system should be designed to handle the pressure for which it has been specified to handle.
  - Tactics

        * Publish contracts
        * Support pluggable adapters

- Cost
    - All software used should not be proprietary but rather open source so as to minimise cost as much as possible.
    - Tactics
        * List of open source software to follow..

- Usability
    - The interface should be lightweight.
    - The interface should be intuitive to use as well as obey Human Computer Interaction guidelines so that it is efficient and easy to use.
    - Tactics
        * Separate user interface
        * User initiative(Cancel, undo)
        * System initiative(Maintain user,system and task models)