Integration architecture requirements

Accessibility challenges

1. Representational State Transfer (REST)
    - REST is a hybrid style derived from several of the network-based architectural styles, which will be suitable for space buzz since it will be web based.
    - Understanding of the system context approach: Rest allows designers to start with the system needs as a whole, with no constrains and incrementally identifies and applies constraints to elements of the system in order to differentiate the design space.
    - Allow the forces that influence system behaviour to flow naturally, in harmony with the system.
    - This is ideal for the system that is designed by different designers concurrently to work efficiently.
2. Protocols:
    - HTTPS for secure connection – this will ensure that student's details and discussions are kept private from intruders.
    - FTP – Lectures will be able to export the report file.
    - SMTP - Lecturers will communicate with each other and students through the space buzz email system.
    - IMAP – an alternative to SMTP.
    - SSL – secure shell in the background.

Infrastructural constraints

1. NULL style
    - The null style describes a system in which there are no distinguished boundaries between components.
2. Uniform interface
    - degrades efficiency, since information is transferred in a standardized form rather than one which is specific to an application's needs
3. Stateless
    - State of a session must be kept and restricted only to the client.
    - Every request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server.
4. Client-server
    - Separation of concerns is the principle behind the client-server constraints.
5. Catchable
    - Most interactions will be eliminated thus improving efficiency, scalability, and user-perceived performance by reducing the average latency of a series of interactions.
    - Although it decrease reliability if stale data within the cache differs significantly from the data that would have been obtained had the request been sent directly to the server.
6. Layered System

- allows an architecture to be composed of hierarchical layers by constraining component behaviour such that each component cannot "see" beyond the immediate layer with which they are interacting

7. Code on Demand
    - Allows client functionality to be extended by downloading and executing code in the form of applets or scripts.
    - But it also reduces visibility.

Quality requirement challenges

1. Performance
2. Scalability
3. Reliability
4. Security
5. Auditability
6. Maintainability

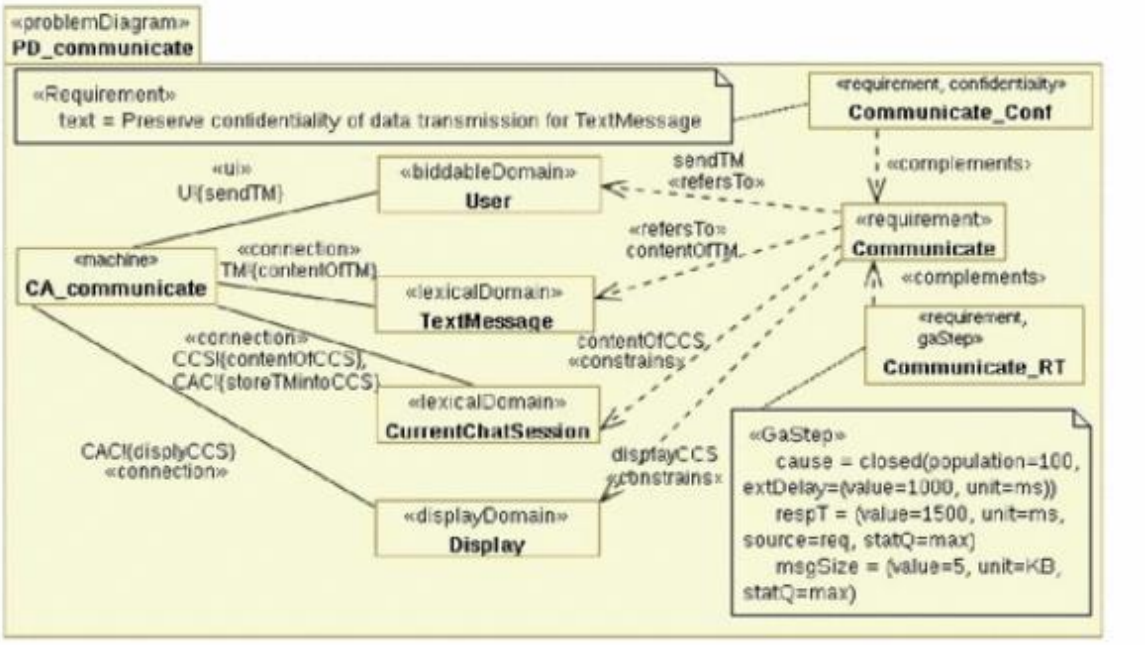

Fig. 1. Problem diagram for the requirement *Communicate*