Mentre nel System Design:intro abbiamo identificato gli obiettivi di progettazioni e decomposto il sistema, in questo capitolo presentiamo le attività di progettazione del sistema che affrontano gli obiettivi di progettazione.
In particolare modo:

- Selezione di componenti standard e legacy: i componenti standard o legacy realizzano sottosistemi specifici in modo più economico. La decomposizione iniziale del sottosistema viene adattata per adattarli.
- Mappatura del sottosistema sull'hardware: quando il sistema viene distribuito su più nodi, sono necessari sottosistemi aggiuntivi per risolvere problemi di affidabilità o prestazioni.
- Progettazione di un'infrastruttura di gestione dei dati persistenti
- Specifica di una politica di controllo dell'accesso: gli oggetti condivisi sono protetti in modo che l'accesso dell'utente sia controllato.
- Progettazione del flusso di controllo globale: la determinazione della sequenza delle operazioni influisce sull'interfaccia dei sottosistemi.
- Gestione delle condizioni limite: una volta identificati tutti i sottosistemi, gli sviluppatori decidono l'ordine in cui i singoli componenti vengono avviati e arrestati.

Quando vi sono delle decisioni che richiedono dei compromessi, gli sviluppatori dividono il sistema in parti gestibili per gestire la complessità: ogni sottosistema è assegnato a un team e realizzato in modo indipendente.
Perché questo sia possibile devono far fronte a determinate scelte quando il sistema viene decomposto:

1. Mapping Hardware/Software:
    1. Qual'è la configurazione hardware del sistema?
    2. Quale nodo è responsabile di una certa funzionalità?
    3. Come è gestita la comunicazione tra i nodi realizzati?
    4. Quali servizi sono realizzati utilizzando componenti software esistenti?
    5. Come queste componenti sono incapsulate?

affrontare tali problemi, può portare alla definizione di sottosistemi aggiuntivi che consentono di trasferire informazioni da un nodo all'altro. Le componenti dovrebbero essere incapsulate per minimizzare le dipendenze da una particolare componente. (se una azienda in futuro fornirà un prodotto migliore o più economico potremo voler cambiare senza influenzare il resto del sistema).

2. Gestione dei dati:
    1. Quale dovrebbe essere l'informazione persistente?
    2. Dove dovrebbe essere memorizzata?
    3. Come accede ai dati persistenti?

I dati costituiscono uno dei fattori molto importanti in quanto rappresentano un "collo di bottiglia": se l'accesso ai tali dati è lento, di conseguenza il sistema si comporterà in un modo lento; se tali dati sono danneggiati, di conseguenza ci sarà una probabilità dell'errore di sistema completo.

3. Controllo dell'accesso:
    1. Chi può accedere alle informazioni?
    2. Il controllo di accesso può cambiare dinamicamente?
    3. Come è specificato e realizzato, il controllo di accesso?

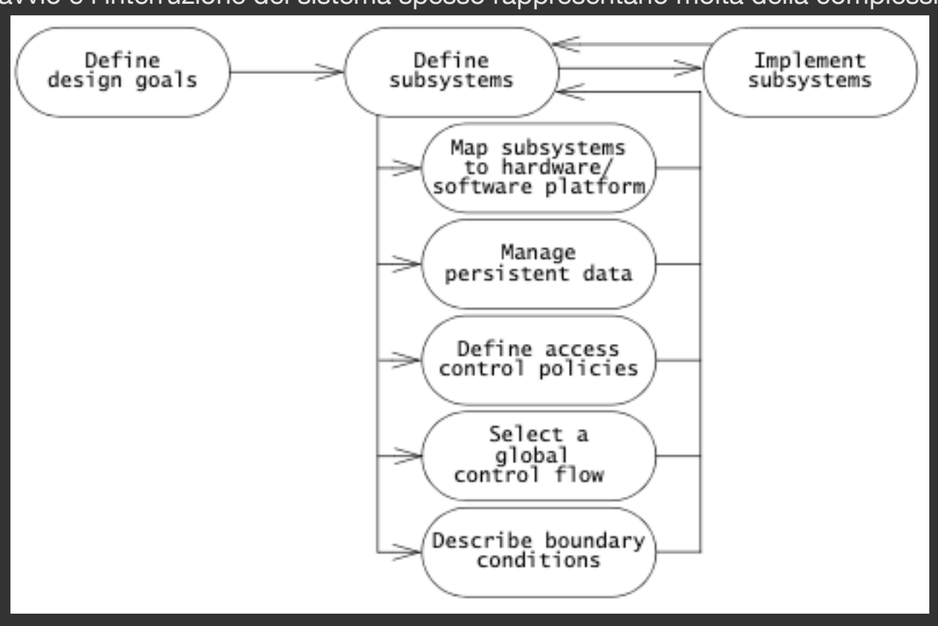
La politica utilizzata per specificare chi può e chi non può accedere a certe informazioni dovrebbe essere la stessa per tutti i sottosistemi.

4. Flusso di controllo:
    1. Come è gestita la sequenza delle operazioni?
    2. Il sistema è guidato da eventi?
    3. Il sistema può gestire più di un'interazione utente alla volta?

La scelta del flusso di controllo ha un impatto sulle interfacce dei sottosistemi. Se viene selezionato un flusso di controllo guidato da eventi, i sottosistemi forniranno un gestore degli eventi. Se vengono selezionati i thread, i sottosistemi devono garantire l'esclusione reciproca nelle sezioni critiche.

5. Boundary Conditions:
    1. Come è avviato il sistema?
    2. Come è interrotto?
    3. Come sono individuati e gestiti i casi eccezionali?

L'avvio e l'interruzione del sistema spesso rappresentano molta della complessità di un sistema, specialmente nei sistemi distribuiti.



SDD: System Design Document
La progettazione del sistema è documentata nel System Design Document. Descrive gli obiettivi di progettazioni stabiliti dal progetto, la decomposizione del sottosistema(con diagrammi di classe UML), la mappatura hardware/software (con diagrammi di distribuzione UML), la gestione dei dati, il controllo degli accessi, i meccanismi del flusso di controllo e le condizioni boundary.
L'SSD viene utilizzato per definire le interfacce tra i team di sviluppatori e fungere da riferimento quando è necessario rivedere le decisioni a livello di architettura.

1. Introduction: fornisce una breve panoramica dell'architettura del software e degli obiettivi di progettazione. Fornisce inoltre riferimenti ad altri documenti e informazioni sulla tracciabilità.
    1. Purpose of the system;
    2. Design Goals;
    3. Definitiion,acronyms, and abbreviations;
    4. References;
    5. Overview;
2. Current software architecture: descrive l'architettura del sistema da sostituire. Se non esiste un sistema precedente, questa sezione può essere sostituita da un rilevazione delle architetture attuali per sistemi simili. Lo scopo di questa sezione è di rendere esplicite le informazioni di base utilizzate dagli architetti di sistema e le questioni comuni che il nuovo sistema affronterà.
3. Proposed software architecture
    1. Overview;
    2. Subsystem decomposition;
    3. Hardware/software mapping;
    4. Persistent data management;
    5. Access control and security;
    6. Global software control;
    7. Boundary conditions;
4. Subsystem services

Glossary.