Техники (тактики) за постигане на качеството на софтуера

Архитектурни тактики: Фундаментални арх. решения, чрез които се контролират резултатите на даден сценарий за качество. Наборът от конкретни тактики се нарича архитектурна стратегия.

1. Тактики за изправност (Dependability): Съвкупност от няколко качества на софтуерните системи, включително Availability, Reliability, Safety, Integrity, Confidentiality, Maintainability. Вкл.:

-Откриване на откази

1. **Ping/Echo**: Компонент А изпраща сигнал до компонент Б и очаква отговор в рамките на определен интервал от време.
2. **Heartbeat/Keepalive**: Компонент периодично излъчва сигнал, който друг компонент очаква.
3. **Exceptions**: Обработка на изключения, генерирани при определено състояние на отказ.

-Отстраняване на откази

1. **Active Redundancy (Hot Restart)**: Дублиране на важни компоненти, които се поддържат в едно и също състояние.
2. **Passive Redundancy (Warm Restart)**: Основният компонент реагира на събитията и информира резервните за промяната на състоянието.
3. **Spare**: Поддръжка на резервни изчислителни мощности, които се инициализират при отказ на компонент.

-Разнородност (Diversity)

1. **Design Diversity**: Използване на разл. програмни езици, компилатори, алгоритми.
2. **Data Diversity**: Различни данни за тестове.
3. **Temporal Diversity**: Изпълнение на програмата в разл. моменти от време.

2. Тактики за производителност (Performance) **Цел**: Постигане на реакция от страна на системата на зададено събитие в рамките на определени времеви изисквания.

-Намаляване на изискванията

1. **Увеличаване на производителността на изчисленията**: Подобряване на алгоритмите, кеширане.
2. **Намаляване на режийните (overhead)**: Избягване на изчисления, които не са свързани с конкретното събитие.
3. **Промяна на периода**: Редуциране на честотата на периодични събития.
4. **Промяна на тактовата честота**: Пропускане на някои събития.
5. **Ограничаване на времето за изпълнение**: При итеративни алгоритми.
6. **Опашка с краен размер**: Заявките, които не могат да се обработят веднага, се поставят в опашка.

-Управление на ресурсите

1. **Паралелна обработка**: Оптимизация на времето чрез паралелна обработка на заявки.
2. **Излишък на данни/процеси**: Cache, load-balancing.
3. **Включване на допълнителни ресурси**: Повече и по-бързи процесори, памет, диск, мрежа.

-Арбитраж на ресурсите

1. **Scheduling**: Приоритизиране на събитията и предаване на управлението на високо-приоритетните събития.

- **FIFO**: Всички заявки са равноправни и се обработват подред.
- **Fixed Priority**: Заявките се обработв. по реда на техн. риоритет.
- **Dynamic Priority**: Последователно или според изискванията за навременност.