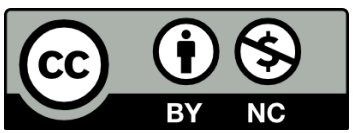
9 DESIGN TACTICS

course “software requirements and architecture”

Nov 2022



contents

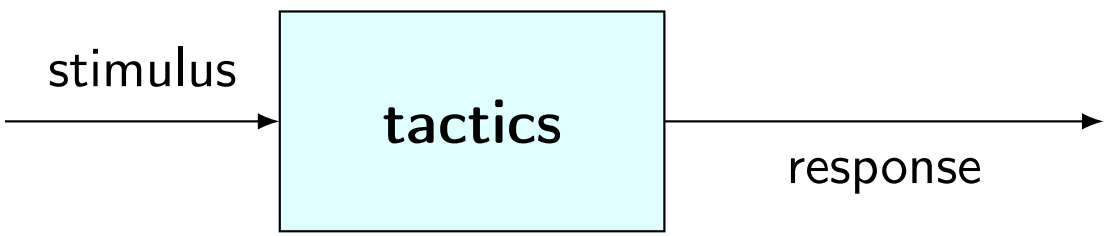
contents

introducing tactics

- Architects need techniques to achieve particular qualities.
- The quality requirements specify the responses of the software system to realize business goals.
- Tactics are used by the architect to create a design.
- Tactics connect quality attribute requirements with architectural decisions.

introducing tactics

- A tactic is a design decision that impacts on specific quality attributes.
- A system design consists of a collection of decisions.
- Some of these decisions help control the quality attribute responses.
- Others ensure achievement of system functionality.
- Each tactic is a design option for the architect.
- For example, one tactic can introduce redundancy to increase the availability of a system.
- This is one possible option, among many, for the architect to increase availability.



types of tactics

1. availability tactics
2. modifiability tactics
3. performance tactics
4. security tactics
5. testability tactics
6. usability tactics

contents

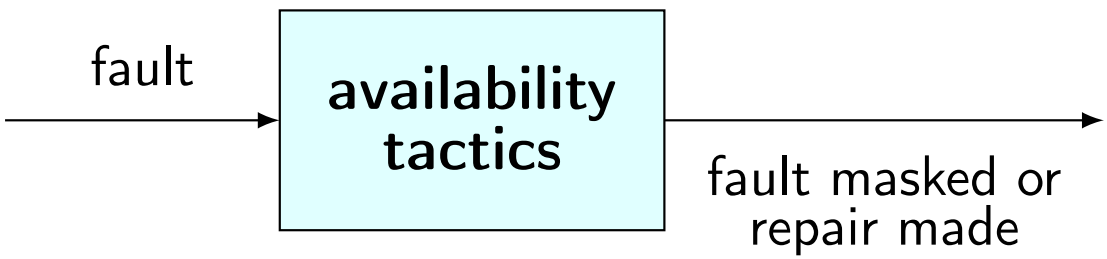
Availability

Availability quantifies the percentage of time during which a given system is operational and working correctly.

- It is an important aspect that has ramifications in other issues: confidence of the users in the products that they use, value of the information, processes efficiency, productivity of the organisations.
- Availability is normally measured by the MTBF (mean time between failures) and MTTR (mean time to repair).
- $\text{availability} = \frac{\text{MTBF}}{\text{MTBF}+\text{MTTR}}$

failure

- A failure occurs when the system no longer delivers a service that is consistent with its specification.
- This failure is observable by the users.
- A (combination of) fault(s) has the potential to cause a failure.
- Recovery or repair is an important aspect of availability.
- Availability tactics aim to avoid faults from becoming failures or at least reduce the effects of the fault and make repair possible.



availability

- Many of the tactics are available within standard execution environments such as operating systems, application servers, and database management systems.
- All approaches to addressing availability involve some type of:
  1. redundancy,
  2. health monitoring to detect a failure,
  3. recovery when a failure is detected.
- The monitoring or recovery is either automatic or manual.

fault detection

- Ping/echo: One component issues a ping and expects to receive back an echo from the component under scrutiny.
- It can be used by clients to ensure that a server object and the communication path to the server are operating.
- Heartbeat: One component emits a heartbeat message periodically and another component listens for it.
- If the heartbeat fails, the originating component is assumed to have failed and a fault correction component is notified.

- Exceptions: One method for detecting faults is to encounter an exception, which is raised when one of the fault classes is recognized.

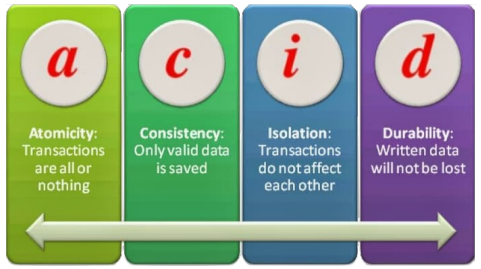
fault recovery

- Voting: Processes running on redundant processors each take the input and compute the output that is sent to a voter.
- Popular voting algorithm are "majority rules" or "preferred component".
- Voting is used to correct faulty operation of algorithms or failure of a processor.

- Active redundancy: All redundant components respond to events in parallel.
- The response from only one component is used (usually the first to respond); the rest are discarded.
- Passive redundancy: One component (the primary) responds to events and informs the other components (the standbys) of state updates they must make.
- When a fault occurs, the system must first ensure that the backup state is sufficiently fresh before resuming services.

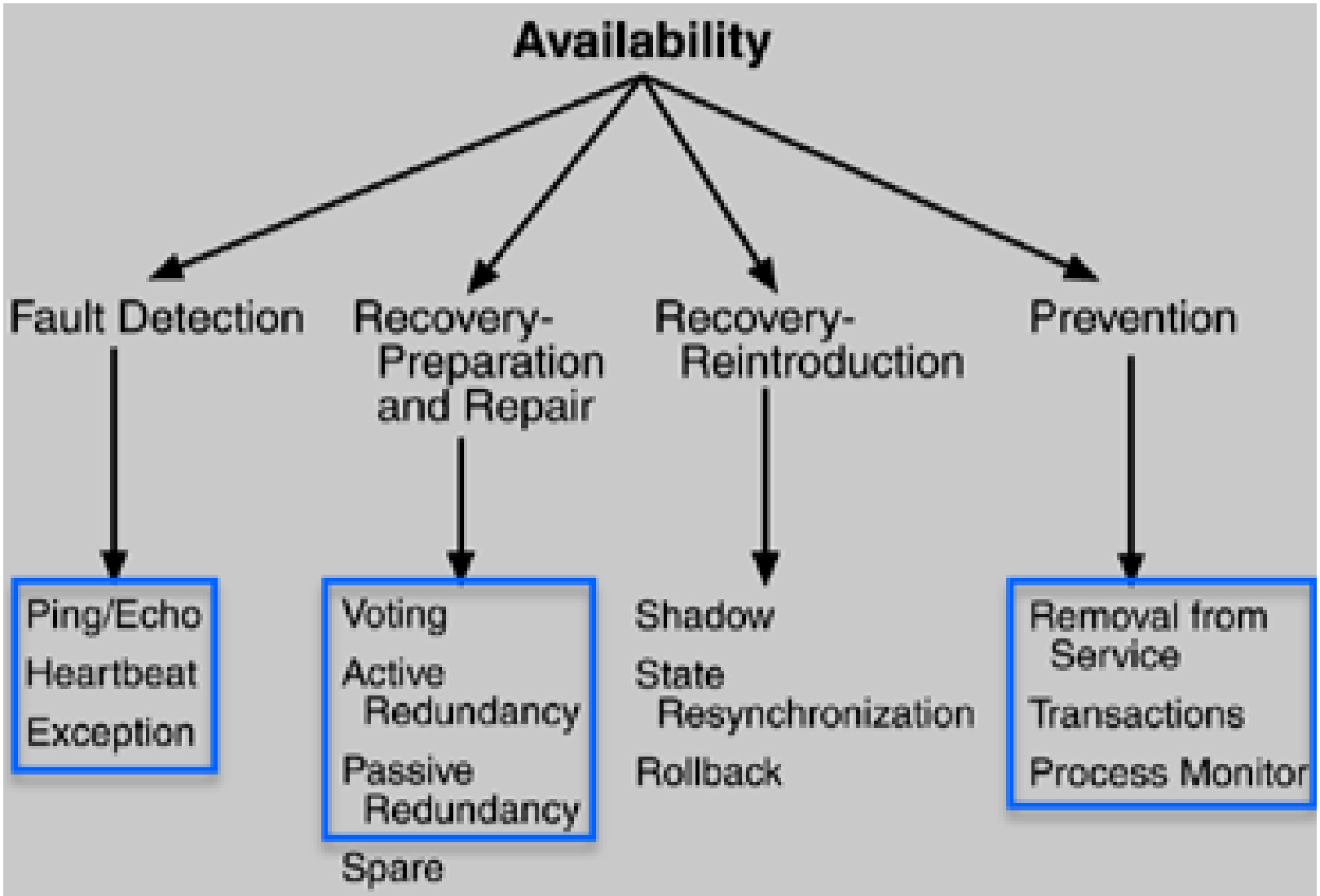
fault prevention

- Removal from service: This tactic removes a component from operation to undergo activities that prevent failures.
- Example: rebooting a component to prevent memory leaks to cause a failure.
- Transactions: A transaction is the bundling of several sequential steps such that the bundle can be undone at once.



- Transactions are used to prevent any data from being affected if one step in a process fails
- They also prevent collisions among several simultaneous threads accessing the same data.
- Process monitor: Once a fault in a process has been detected, a monitoring process can delete the nonperforming process.
- A new instance of the process needs to be created.

availability



contents

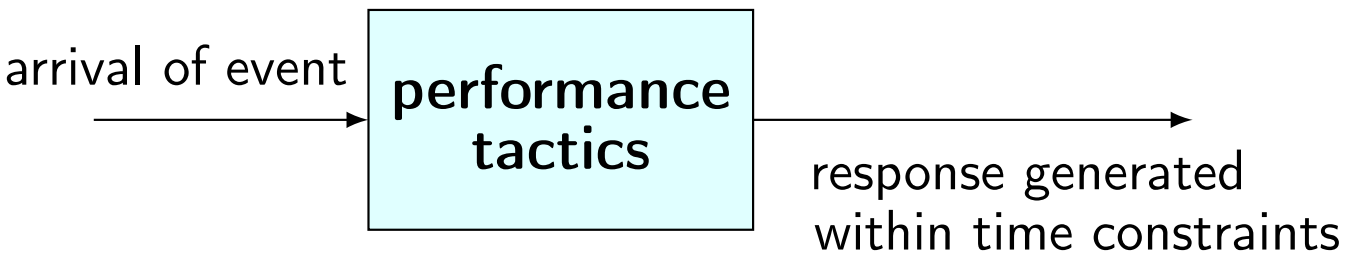
Performance

Performance refers to the capacity of a system to respond to its stimulus, that is, the time necessary for responding to the events or the number of events processed by time unit.

- Google loses 20% traffic if their web sites respond 500 ms slower than usual.
- Amazon loses 1% of revenue for every 100 ms in latency.
- A Mozilla study shows that if the webpage is not loaded within 1 to 5 seconds, users will leave it.

events

- Performance tactics generate a response to an event arriving at the system within some time constraint.
- The event can be single or a stream and is the trigger for a request to perform computation.
- It can be the arrival of a message, the expiration of a time interval, the change of state in the system environment, etc.
- The system processes the events and generates a response.
- Performance tactics control the time within which a response is generated.
- Latency is the time between the arrival of an event and the generation of a response to it.



resource consumption and blocked time

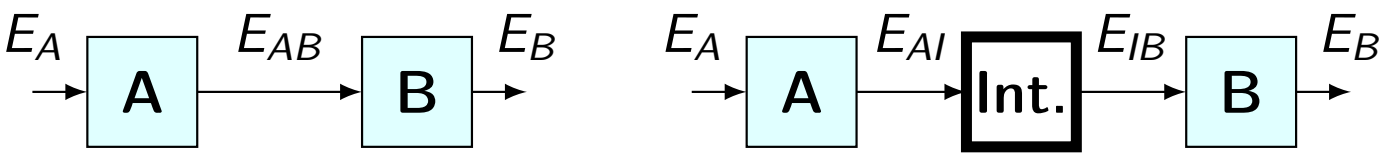
- After an event arrives, either the system is processing on that event or the processing is blocked for some reason.
- Resource consumption: Resources (CPU, data stores, network bandwidth, memory, and buffers) must be managed.
- Access to critical sections must be made sequential.
- All the phases to process an event contribute to the overall latency of the processing of that event.
- Blocked time: A computation can be blocked from using a resource due to its contention or unavailability.
- Three tactic categories address performance:
  1. resource demand
  2. resource management
  3. resource arbitration

resource demand

- Event streams are the source of resource demand.
- Two main aspects in resource demand are:
  1. The time between events in a resource stream
  2. How much of a resource is consumed by each request
- One tactic to decrease latency is to reduce the resources required for processing an event stream.
- Increase computational efficiency: One step in the processing of an event is applying some algorithm.
- Improving the algorithms decreases latency.
- Sometimes one resource can be traded for another.
- For example, data may be kept in a local/fast repository to avoid getting it from a slow resource.

resource demand

- Reduce computational overhead: If there is no request for a resource, processing needs are reduced.
- The use of intermediaries[1] (important for modifiability) increases the resources consumed in processing an event stream, and so removing them improves latency.



- This is a modifiability/performance tradeoff.

[1] If module A has a dependency on module B, one can insert an intermediary between A and B to manage the activities associated with the dependency. The intermediary breaks the dependency.

resource demand

- Another tactic for reducing latency is to reduce the number of events processed.
- Manage event rate: If one can reduce the sampling frequency at which variables are monitored, demand can be decreased.
- This is often possible if the system was overengineered.
- Control frequency of sampling: If the arrival of externally generated events is not controlled, queued requests can be sampled at a lower frequency, possibly resulting in the loss of requests.

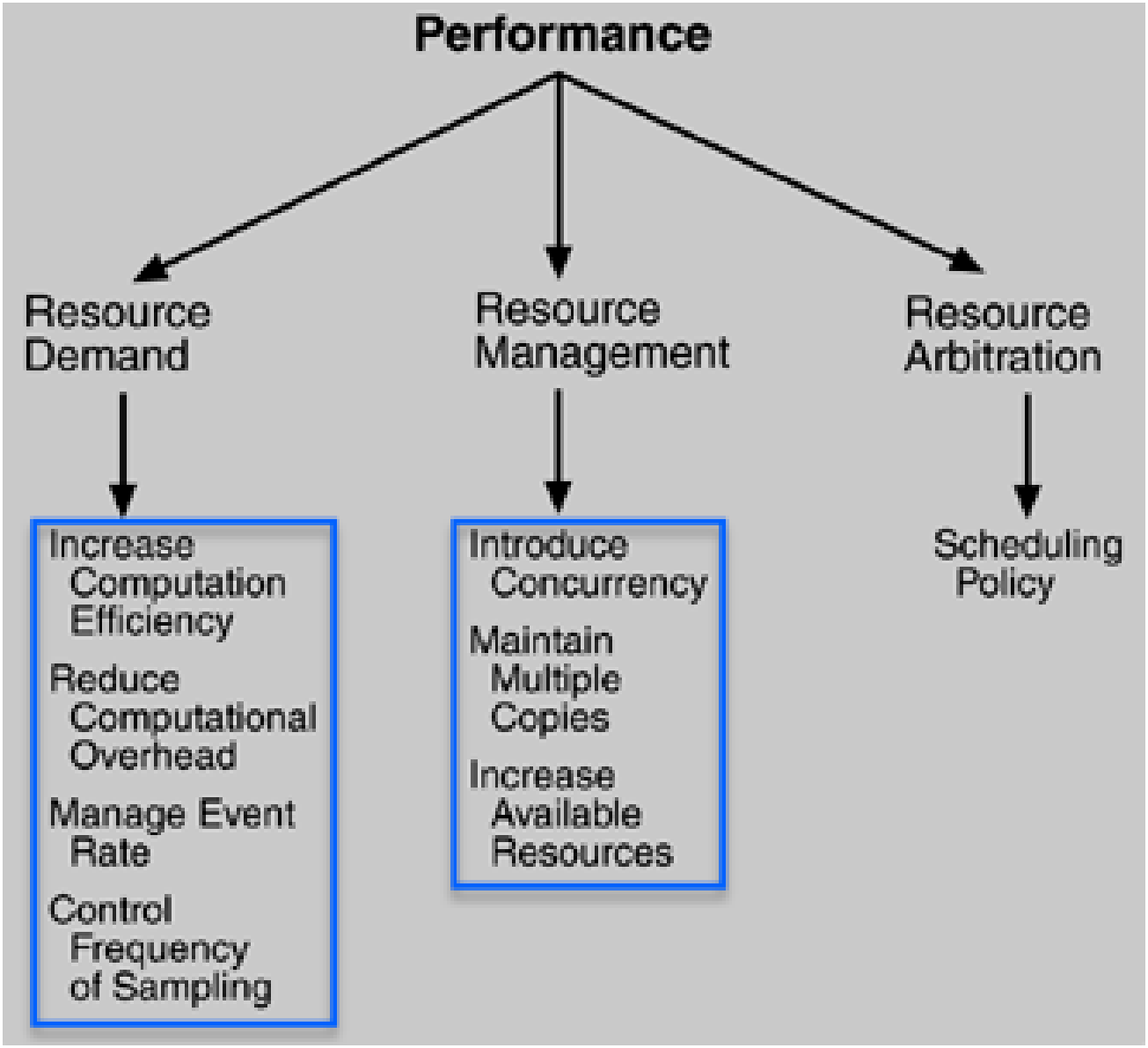
resource management

- Introduce concurrency: If requests can be processed in parallel, the blocked time can be reduced.
- Concurrency can be introduced by processing different streams of events on different threads.
- Appropriately allocating the threads to resources (load balancing) is important to maximally exploit the concurrency.
- Maintain multiple copies of either data or computations: Clients in a client-server pattern are replicas of the computation.
- The purpose of replicas is to reduce the contention that would occur if all computations took place on a central server.

resource management

- Caching is a tactic to reduce contention, in which data is replicated, either on different speed repositories or on separate repositories.
- Since the cache data is a copy of existing data, it is relevant to keep the copies consistent and synchronized.
- Increase available resources: Faster processors, additional processors, additional memory, and faster networks all have the potential for reducing latency.
- Cost is usually a consideration in the choice of resources.
- This is a cost/performance tradeoff.

performance



Summary

- Tactics are used by the architect to create a design.
- A tactic is a design decision that impacts on specific quality attributes.
- There are tactics for availability, modifiability, performance, security, testability, and usability.
- Availability tactics try to keep faults from becoming failures or at least reduce the effects of the fault and make repair possible.
- Performance tactics generate a response to an event arriving at the system within some time constraint.

bibliography

- Bass L, Clements P, and Kazman R; *Software Architecture in Practice*, 2nd edition, Addison-Wesley, 2003. [chapter 5]