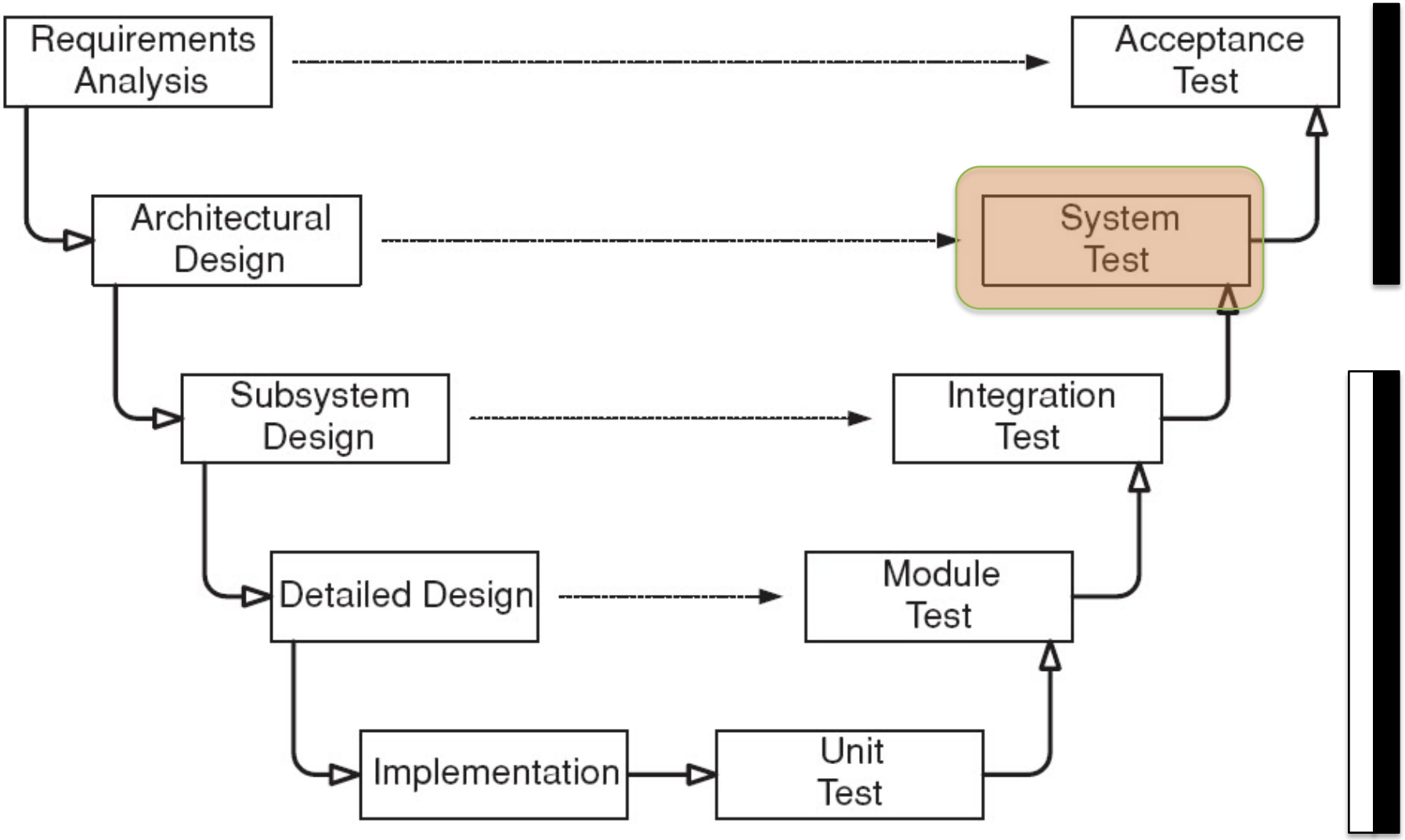
Functional Testing

Input space definition

Andrea Caracciolo

Top-down


Requirements Analysis
Acceptance Test
Architectural Design
System Test
Subsystem Design
Integration Test
Detailed Design
Module Test
Implementation
Unit Test

Functional testing

1. Identify parameters
2. Identify parameter characteristics
3. Identify representative values
4. Generate test-case specifications
5. Generate test-cases
6. Run test cases

- The maximal price of the bid must be >= the minimal price defined in the auction.
- The maximal price of the bid can be increased anytime, as long as the auction is not over.
- Two bidders can not have the same maximal price.
- If the n>0 bidders are sorted according to their maximal prices MP1 <= MP2 <= ... <= MPn, the current winner of the auction is the bidder n; the current selling price is defined as MP(n-1)+increment. The current winner and the current selling price is updated each time a new bid is placed or increased. If there are no bidder (n=0), there is no current winner nor current selling price.
- Buyers see the item minimal price, increment, current winner, and current selling price.
- When the current winner changes, the old winner receives an email.
- At the time the auction terminates and there was at least one bidder (n>0), the transaction proceeds in accordance with the current winner and the current selling price at that time. The winner receives a confirmation email. If there was no bidder (n=0), the auction simply closes.
- The highest bidder can't use the money they bid as long as the auction is not over. You need to ensure, I'm able to pay the bid.
- If several users interact with the system concurrently, a user might see and act on stale (outdated) data, e.g. stale selling price. The system must detect such situation (optimistic locking?) and inform users accordingly.

Functionality: Placing a bid

Parameters

1. **Identify parameters**
   - Independently testable features
   - Other elements of the environment on which the unit depends on
     - E.g. database, application state, ..
2. **Identify parameter characteristics**
   - Meaningful attributes for each parameter

• The maximal price of the bid must be >= the minimal price defined in the auction.
• The maximal price of the bid can be increased anytime, as long as the auction is not over.
• Two bidders can not have the same maximal price.
• If the n>0 bidders are sorted according to their maximal prices MP1 <= MP2 <= ... <= MPn, the current winner of the auction is the bidder n; the current selling price is defined as MP(n-1)+increment. The current winner and the current selling price is updated each time a new bid is placed or increased. If there are no bidder (n=0), there is no current winner nor current selling price.
• Buyers see the item minimal price, increment, current winner, and current selling price.
• When the current winner changes, the old winner receives an email.
• At the time the auction terminates and there was at least one bidder (n>0), the transaction proceeds in accordance with the current winner and the current selling price at that time. The winner receives a confirmation email. If there was no bidder (n=0), the auction simply closes.
• The highest bidder can't use the money they bid as long as the auction is not over. You need to ensure, I'm able to pay the bid.
• If several users interact with the system concurrently, a user might see and act on stale (outdated) data, e.g. stale selling price. The system must detect such situation (optimistic locking?) and inform users accordingly.

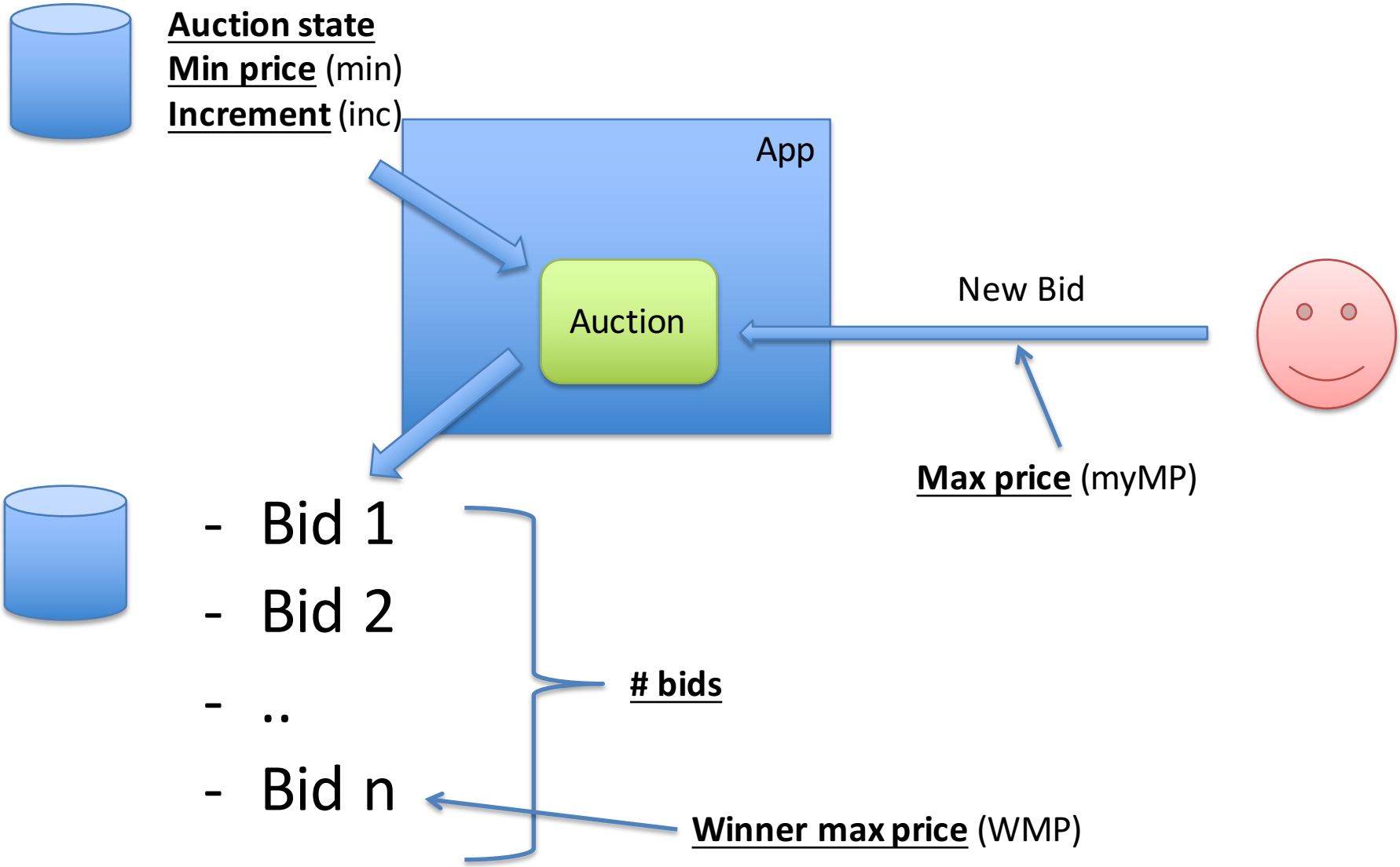
- The maximal price of the **bid** must be >= the minimal price defined in the **auction**.
- The maximal price of the bid can be increased anytime, as long as the *auction is not over*.
- Two bidders can not have the same maximal price.
- If the n>0 bidders are sorted according to their maximal prices MP1 <= MP2 <= ... <= MPn, the current winner of the auction is the bidder n; the current selling price is defined as MPn+increment. The current winner and the current selling price is updated each time a new bid is placed or increased. If there are no bidder (n=0), there is no current winner nor current selling price.
- Buyers see the item minimal price, increment, current winner, and current selling price.
- When the current winner changes, the old winner receives an email.
- At the time the auction terminates and there was at least one bidder (n>0), the transaction proceeds in accordance with the current winner and the current selling price at that time. The winner receives a confirmation email. If there was no bidder (n=0), the auction simply closes.
- The highest bidder can't use the money they bid as long as the auction is not over. You need to ensure, I'm able to pay the bid.
- If several users interact with the system concurrently, a user might see and act on stale (outdated) data, e.g. stale selling price. The system must detect such situation (optimistic locking?) and inform users accordingly.

Red: parameters
Blue: paramenter characteristics
Grey: side effects / outcome

Parameters

- Bid
  - Max price
- Auction
  - Increment
  - Min price
  - # bids
  - Winner max price
  - Auction state

Parameters


Auction state
Min price (min)
Increment (inc)
App
Auction
New Bid
Max price (myMP)
- Bid 1
- Bid 2
- ..
- Bid n
bids
Winner max price (WMP)

Values

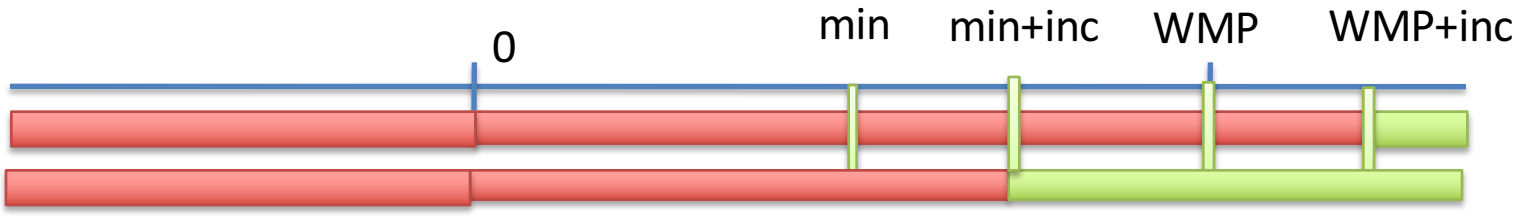
3. Identify representative values

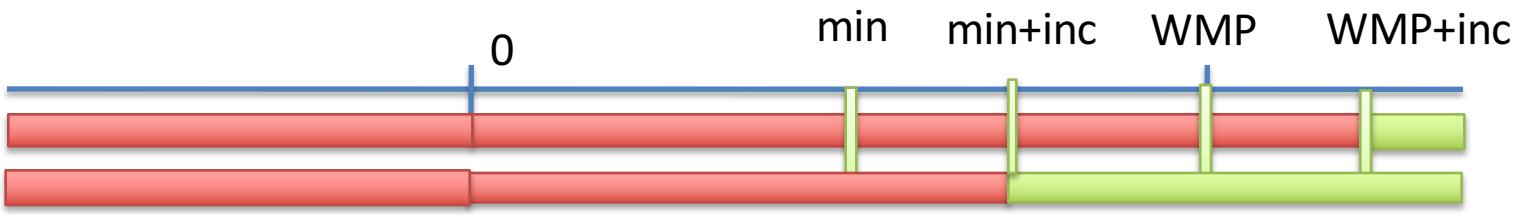
– *equivalent partitions*

Days in a month (example):

- Valid: 1 <= X <= 28
- Valid: 28 < X <= 31 [if month is ..]
- Invalid: X < 1, X > 12

myMP :





- Auction (context)
  - Increment (inc): > 0
  - Min price (min): 0, > 0
  - # bids (bids): 0, > 1
  - Winner max price (WMP): NO, >= min+inc
  - Auction state (state): running, closed, not existing

Specification

4. **Generate test-case specifications**
   - Check boundary values : A <= X <= B
     - Invalid: A – 1, B + 1
     - Boundary: A, B
     - Valid: A < n < B

Specification

4. Generate test-case specifications

- Bids = 0
  - Invalid: min; min+inc-1
  - Boundary: min+inc
  - Valid: min+inc+1
- Bids > 0
  - Invalid: min; WMP; WMP+inc-1
  - Boundary: WMP+inc
  - Valid: WMP+inc+1
- * [error if status != running]