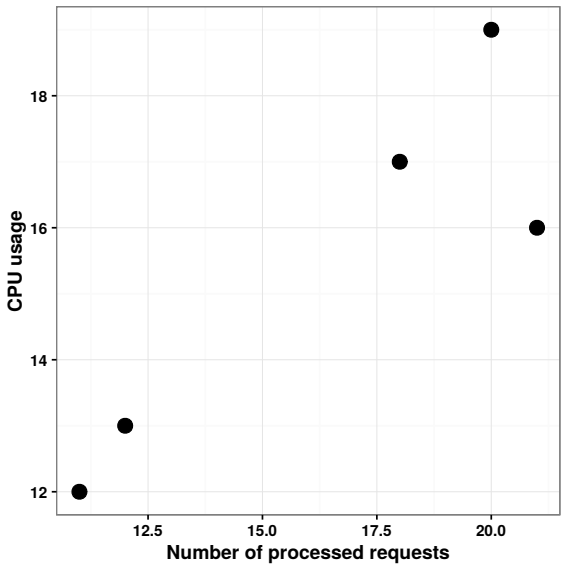
# 6th Seminar – Requirements Analysis, Explorative Data Analysis – Solutions

## 1 Exploratory data analysis of server performance

We measured the following performance metrics on a server:

| Time of measure [ms] | 500 | 600 | 700 | 800 | 900 |
|---|---|---|---|---|---|
| Requests processed in the last 100 ms [request] | 11 | 12 | 21 | 18 | 20 |
| Average serving time in the last 100 ms [ms] | 15 | 20 | 21 | 25 | 27 |
| CPU utilization of the last 100 ms [%] | 12 | 13 | 16 | 17 | 19 |
| HDD I/O utilization of the last 100 ms [%] | 55 | 63 | 87 | 61 | 73 |

a. Display the number of processed requests and the CPU utilization on a scatterplot diagram! Interpret the diagram!
**Solution**




We can see two clusters (groups) with roughly positive correlation (values are proportional more or less) but it's not entirely monotonic (something else may affect the values, that's why they vary). Interpreting the result: the utilization of the CPU grows with the number of processed requests. The left bottom cluster contains the moments with low loads, while the right upper cluster contains the moments with higher loads.This observation could be a good starting point when analysing the load (like the moments in the same cluster are close to each other or not).

b. What is the server's throughput at the time of the first measure? What is the average and median throughput based on these 5 measurements? What belongs to the 40% quantile?
**Solution**

We can see from the times of the measures that 100 ms elapsed between two measure. From this:

$$X_1 = \frac{r_1}{\Delta t} = \frac{11 \text{ request}}{100 \text{ ms}} = \frac{11 \text{ request}}{100 \text{ ms}} \left[\frac{1000 \text{ ms}}{1 \text{ s}}\right] = 110 \frac{\text{request}}{\text{s}}.$$

We can calculate the average throughput by either calculating the 4 other throughputs or by using the observation that $\Delta t$ is always 100ms (and using the average number of requests):

$$\bar{r} = \frac{\sum_{i=1}^{n} r_i}{n} = \frac{11 + 12 + 21 + 18 + 20}{5} = 16{,}4$$

From this the average throughput is $\bar{X} = \frac{\bar{r}}{\Delta t} = \frac{16{,}4 \text{ request}}{0{,}1 \text{ ms}} = 164 \frac{\text{request}}{\text{s}}$.
The ordered number of requests are: $11, 12, 18, 20, 21$. From this we can see that the median is 18, so the median of the throughput is $\frac{18 \text{ request}}{0{,}1 \text{ ms}} = 180 \frac{\text{request}}{\text{s}}$.

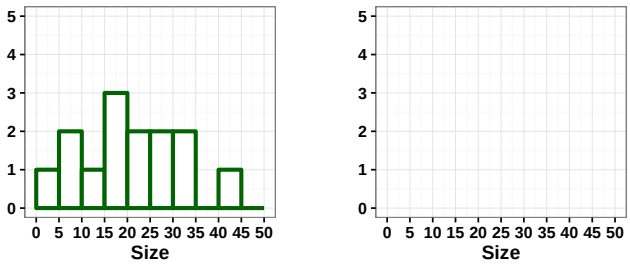
A value of the dataset is called the $p$ quantile if 40% of the values of the dataset is less than or equal to that value. A value is *in* the $p$ quantile if it's less than or equal to the $p$ quantile. A special version of quantile is percentile which uses integer numbers instead of percentages, and the quartile that "splits" the dataset into four parts (quarters). For example the 35. percentile corresponds to the 35% quantile (the quantile could have been for example 35,7%!), and the second quartile corresponds to the 50% quantile (which is the median).
The lower 40% of the processed request values are 11 and 12, so the 40% quantile is 12; the values that are *in* the 40% quantile are 11 and 12. The corresponding throughputs are 110 $\frac{\text{request}}{\text{s}}$ and 120 $\frac{\text{request}}{\text{s}}$.

c. Can we assume some kind of causal relationship between some of the metrics?
**Solution**
As we can see it in the figure the throughput has an effect on the utilization of resources. The high utilization of the bottleneck resource (the HDD – see the performance modelling seminar) corresponds to high response time values.

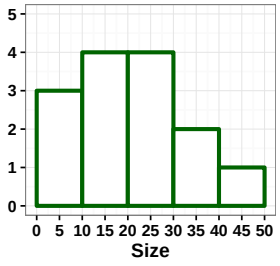
## 2 Picture gallery – data analysis

In our online picture gallery users can search and display pictures that match some search phrase.

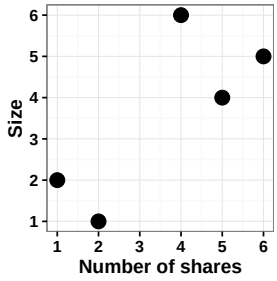
a. We displayed the distribution of the album's size on the following histogram. We would like a histogram with twice the bin size of the original one, since in order to organize our storage more efficiently we only need to know how many albums we have with picture sizes below 10, between 10 and 20, and so on. Create this new histogram!

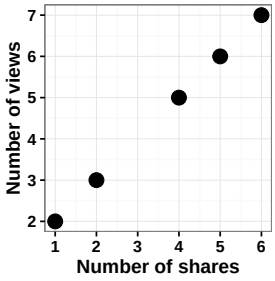


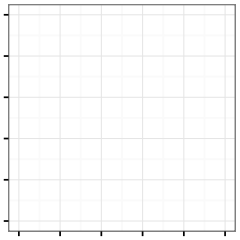

**Solution**
The procedure is the following: we merge the 0–5 and 5–10 bins of the original histogram into a 0–10 bin in the new histogram. The new bin has 3 elements in it, since the original two bins had 1 and 2 elements in them. We repeat this for the 10–15 and 15-20 bins (getting $1 + 3 = 4$ elements), and the next two, etc.



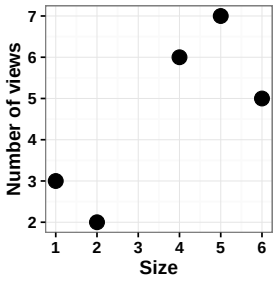

b. We chose 5 albums and displayed their size and number of views in relation to their number of shares on two scatterplot diagrams. Is it true that the bigger is the album, the more view it has? Answer this question with a third scatterplot diagram that shows the albums' number of views in relation to their size!



**Solution**
As we can see in the figure, the statement is not always true, only in the middle part of the diagram.



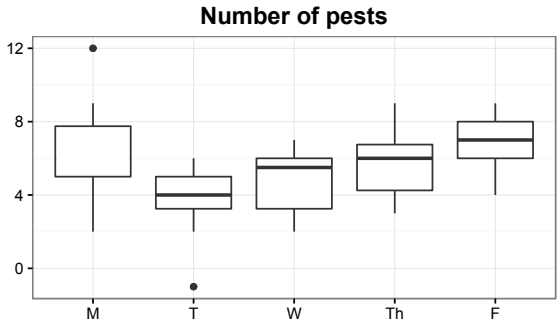

c. We would like to know the popularity of the albums so we calculated the average and median of the number of views based on the scatterplot diagram. Can we always use this method based on a scatterplot diagram? How will these values change if we upload a new album that was viewed 40 times?
**Solution**
The number of views are $2, 3, 5, 6, 7$; so the average is $\frac{23}{5} = 4{,}6$ and the median is 5.
If we add 40 to the set, then the average changes to $\frac{63}{6} = 10{,}5$ while the median stays 5 (since half of the elements are still less than or equal to 5).
*Lesson:* the average is more sensitive to outlier values, while the median is more robust (which means less sensitive).
Can we use this method in general? Not necessarily, since the phenomenon of overplotting could cause some errors in the calculated metrics.

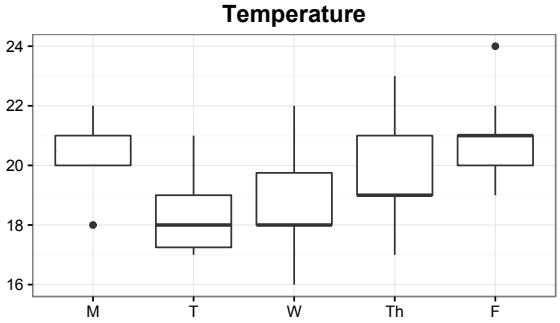
# 3 Sensor network (previous exam exercise) – data analysis

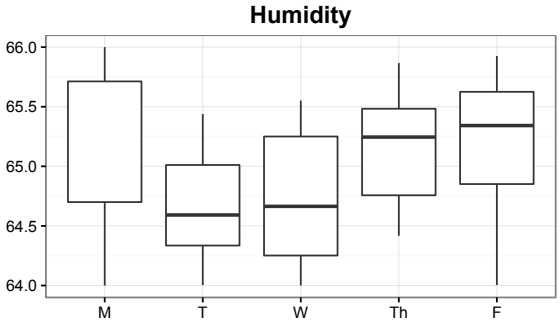
We have an agriculture sensor network that helps us to track the states of our open-field, glasshouse and foil tent areas based on some measured values (temperature, humidity, luminous intensity, wind speed, detected pests, etc.).

| Date | Temp. [°C] | Hum. [%] | Pests [piece] |
|---|---|---|---|
| 2015. 05. 04. 08:00 | 18 | 66,00 | 3 |
| 2015. 05. 04. 09:00 | 20 | 65,75 | 6 |
| 2015. 05. 04. 10:00 | 20 | 65,75 | 8 |
| 2015. 05. 04. 11:00 | 20 | 65,50 | 9 |
| 2015. 05. 04. 12:00 | 20 | 65,50 | 5 |
| 2015. 05. 04. 13:00 | 21 | 65,00 | 12 |
| 2015. 05. 04. 14:00 | 21 | 64,70 | 5 |
| 2015. 05. 04. 15:00 | 21 | 64,70 | 6 |
| 2015. 05. 04. 16:00 | 21 | 64,60 | 7 |
| 2015. 05. 04. 17:00 | 22 | 64,00 | 2 |

a. Unfortunately the middle values (median) of Monday, May 4th are missing from the figures. Draw them based on the data in the table!
b. Interpret the diagrams: which variable's/variables' first quartile is strictly monotonic in time?
c. (Extra task.) We would like to compare the temperature values and pest numbers of Monday in a parallel coordinates diagram.



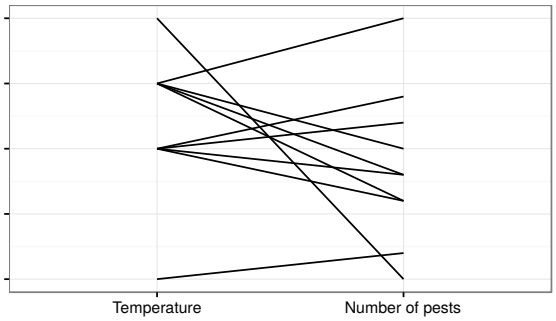



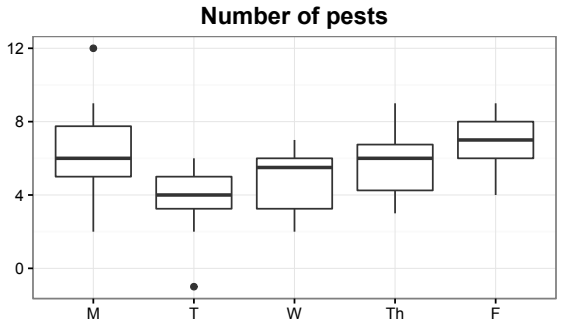

**Solution**

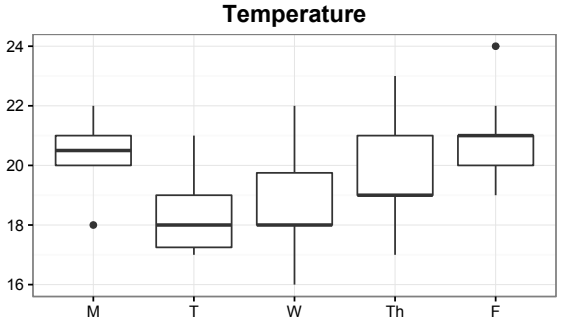
a. Lets draw the medians on the boxplots. Since we have an even number of values, the median will be the smaller of the two middle values. The first two column is ordered, so the medians will be: 20 and 65.
After sorting the third column: $2, 3, 5, 5, 6, 6, 7, 8, 9, 12$, the median is 6.

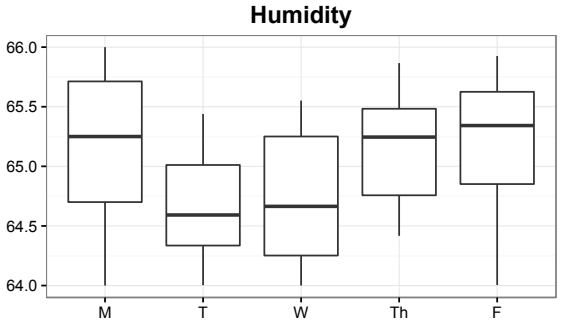
b. None of the variables has this property, since none of the bottoms of the "boxes" show a strictly monotonic change.
The main properties of the boxplot diagram are shown on Figure 1. If a value is outside $\pm 1.5 \times \text{IQR}$ then we display it as a dot.
It's interesting to note that using the constant 1,5 is a statistical convention, which is analogous to the $\pm 3\sigma$ principle of the datasets with normal distribution (see Probability Theory course).

c. The values are displayed in the following parallel coordinates diagram. Looking at the parallel coordinates diagram, we see no strong correlation between the two variables.



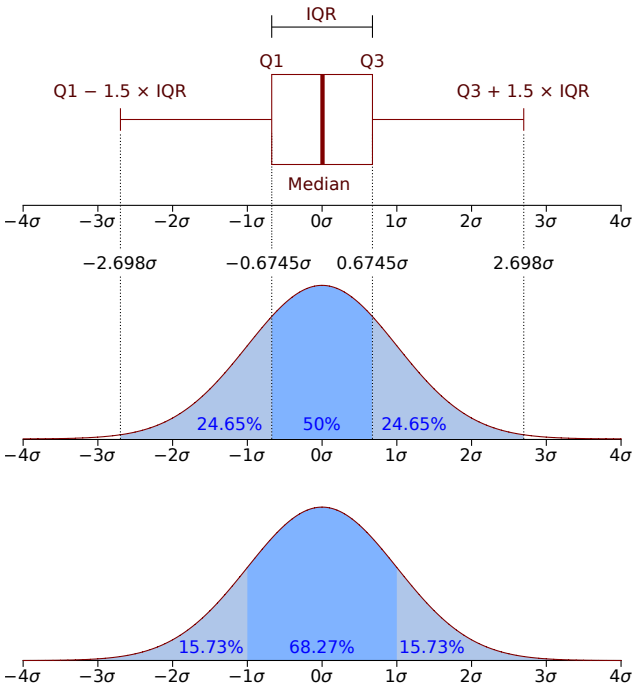




Figure 1: The main properties of the boxplot

# 4 Sensor network (previous exam exercise) – perf. analysis (∗)

(Performance analysis exercises related to Exercise 3.) The different types of sensors provide data from a 100 meters radius around their location. The sensors forward their timestamped data to the central server through a radio communication-based network. The central server processes the requests then archives them to a storage unit. Our organization installed 4500 sensors and each one sends one measurement data in every minute. The system can successfully handle this load. The radio communication network can forward 100 measurement data every second. The central server's CPU is idle (not doing anything) in 75% of the time. Writing a measurement data to the storage unit takes 8 ms.

a. How many measurement data in a second is the current throughput of the system?
**Solution**
Little's law: $X = 4500 \cdot \frac{1 \text{ data}}{60 \text{ s}} = 75 \ \frac{\text{data}}{\text{s}}$

b. What is the throughput, maximum throughput and utilization of the radio network, CPU and storage?
**Solution**
$X_{\text{network}} = X_{\text{CPU}} = X_{\text{storage}} = X = 75 \ \frac{\text{data}}{\text{s}}$, since every visitation number is 1.
$X^{\max}_{\text{network}} = 100 \ \frac{\text{data}}{\text{s}}$ is given in the text $\rightarrow U_{\text{network}} = \frac{X_{\text{network}}}{X^{\max}_{\text{network}}} = 75/100 = 0{,}75 \rightarrow 75\%$
$U_{\text{CPU}} = 1 - 0{,}75 = 0{,}25 = 25\%$ is also given $\rightarrow X^{\max}_{\text{CPU}} = \frac{X_{\text{CPU}}}{U_{\text{CPU}}} = 75/0{,}25 \ \frac{\text{data}}{\text{s}} = 300 \ \frac{\text{data}}{\text{s}}$
$T_{\text{storage}} = 0{,}008$ s and there is no overlap:
$X^{\max}_{\text{storage}} = \frac{1}{T_{\text{storage}}} = 125 \ \frac{\text{data}}{\text{s}} \rightarrow U_{\text{storage}} = \frac{X_{\text{storage}}}{X^{\max}_{\text{storage}}} = 75/125 = 0{,}6 \rightarrow 60\%$

c. How many more sensors can we install (to improve the measurement accuracy) without upgrading our infrastructure? Assume linear scaling!
**Solution**
Since we use all three resources to process the requests:

$$X^{\max} = \min(X^{\max}_{\text{network}}, X^{\max}_{\text{CPU}}, X^{\max}_{\text{storage}}) = X^{\max}_{\text{network}} = 100 \ \frac{\text{data}}{\text{s}}$$

Since the current throughput is $75 \ \frac{\text{data}}{\text{s}}$, a $4:3$ ratio scaling is possible, which means an additional 1500 sensors.

d. The radio network uses smart encoding, so more than one sensor can forward data at the same time. How many sensors are forwarding data at the same time (overlapping) over the network currently and during maximum load, if a forwarding takes 40 ms?
**Solution**
We can use Little's law to calculate the number of messages that are transmitted at the same time:

$$T_{\text{network}} = 0{,}040 \text{ s}$$

$$X_{\text{network}} = 75 \ \frac{\text{data}}{\text{s}} \rightarrow N_{\text{network}} = X_{\text{network}} \cdot T_{\text{network}} = 75 \ \frac{\text{data}}{\text{s}} \cdot 0{,}04 \text{ s} = 3 \text{ messages transmitting currently}$$

$$X^{\max}_{\text{network}} = 100 \ \frac{\text{data}}{\text{s}} \rightarrow N_{\text{network}} = 100 \ \frac{\text{data}}{\text{s}} \cdot 0{,}04 \text{ s} = 4 \text{ messages transmitting maximum}$$

# 5 Requirement analysis of train protection system

We are designing a train protection system. The main goal of the system is to prevent the collision of trains. The key to building a proper system is a requirement specification of good quality, since the test cases and other control mechanisms will be built based on these requirements.

Table 1: Train protection system requirements (partial)

| | | |
|---|---|---|
| R1 | **Safety** | Trains mustn't collide on the supervised track system. |
| R2 | **Operation** | It must be ensured that the trains reach their destination. |
| R3 | **Optimality** | The travel time of trains must be minimized. |
| R4 | **Track sections' supervision** | The track must be divided into sections and maximum one train can be on a section. |
| R5 | **Dividing into sections** | The track must be divided into sections. |
| R6 | **Occupancy** | Maximum one train can be on a section. |
| R7 | **Detecting occupancy** | We have to detect somehow whether there is a train on a section or not. |
| R8 | **Fault tolerance** | We have to be prepared for the malfunction of components. |
| R9 | **Occupancy sensors** | Occupancy of a section must be detected in a redundant manner, based on multiple types of sensors. |
| R10 | **Rail sensor** | Rail sensors must be installed in every section that signal whether there is a train on the section or not. |
| R11 | **Camera system** | Cameras must be installed on the sections where it is possible for observation purposes. |
| R12 | **Positioning** | Trains must continuously signal their positions towards the central control unit. |
| R13 | **GPS subsystem** | The trains must be equipped with a GPS subsystem. |
| R14 | **Wireless connection** | It must be ensured that the trains can provide their positions to the central control unit via a wireless network. |
| R15 | **Train control** | It must be ensured that a train can be stopped before driving onto an occupied section. |
| R16 | **Stopping trains** | The central control unit must be able to immediately stop a train. |
| R17 | **Support of train types** | The system must support every type of trains capable of travelling on rails. |
| R18 | **Unmodifiable trains** | We mustn't use methods that require to change the control system of trains. |

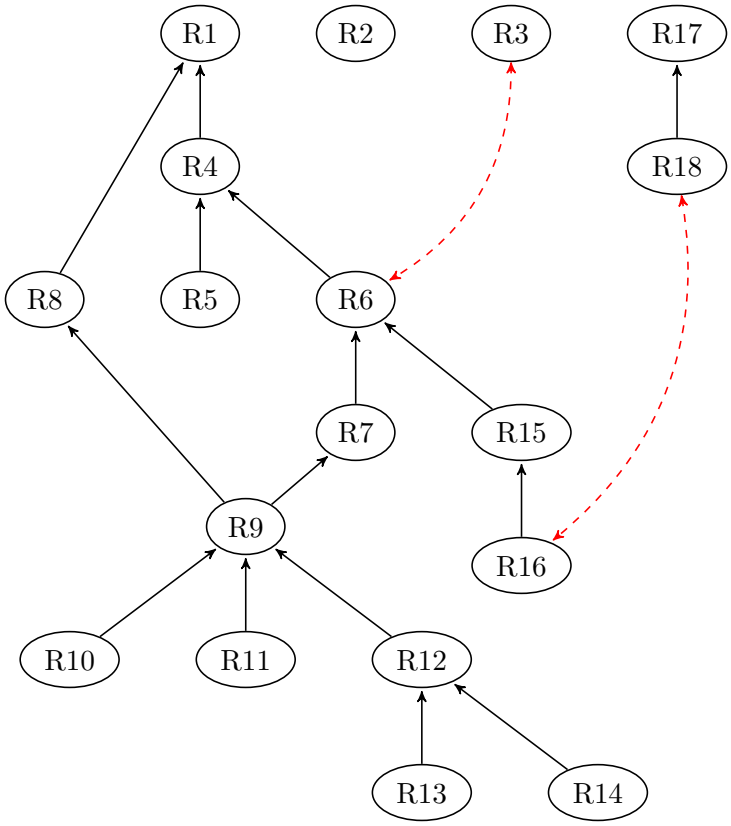
a. Gather the participants that are involved (or affected) when building a system like this (so called *stakeholders*), that is, they can make demands for the system in form of requirements!
**Solution**

- Railway company
  - Railway maintenance
  - Train drivers
  - Traffic controllers
- Train manufacturers
- Passengers
- Authorities
  - Laws and regulations
  - Supervisor boards
- Standards
- etc.

b. After identifying the stakeholders we gathered their requirements, a part of which can be seen in Table 1. Construct a graph that shows the dependencies between the requirements! Draw a directed arc in the graph from $A$ to $B$ if *(1)* requirement $A$ is part of requirement $B$ (*composition*), *(2)* requirement $A$ *refines* requirement $B$, or *(3)* requirement $A$ can be *derived* from requirement $B$. The exact relations between the requirements are not important, only that there is a relation.

**Solution**

The shown relation types reflect the top-bottom thinking of the designer during requirement analysis, so the existence and type of a relation is quite subjective. Here the important goal is to explore the relation (any kind) between higher and lower level requirements. Based on such a hierarchy we can implement the system following the bottom-up method, starting with the low level requirements. Because of this we omit the exact type of relation between the requirements.




(The dashed red lines will come up in a later task.)

c. From the above requirements which are functional requirements? What type of extra-functional requirements can we find in the table (safety, performance, reliability, etc.)?
**Solution**
Functional: R2, R4–R7, R9–R16
Extra-functional: R1 (safety), R3 (”performance”), R8 (reliability), R17 (compatibility), R18 (”maintainability”)
Notice that in order to guarantee an extra-functional requirement we might need to implement a functional (derived) requirement. The converse can happen also: guaranteeing a functional requirement might depend on an extra-functional requirement.

d. Are the gathered requirements consistent? If not, then show an example of a contradiction.
**Solution**
There are two contradictions in the system of requirements (denoted by dashed red arcs on the requirement graph). The contradiction between R3 and R4 (or R6) is easier to resolve. The contradiction originates from allowing only one train on a section which will probably results in a suboptimal travel time (especially if the sections are long, or longer than the trains). Here we need a compromise between safety and efficiency which is also a common occurrence in real life. The contradiction is between R16 and R18 because in order to stop the trains from the central control unit we would probably need to modify their control mechanisms. Resolving this conflict is not about compromises. We need to modify parts of the requirement based on which one is more important.

e. From the gathered requirements give examples for directly verifiable requirements!
**Solution**
For example requirements R6, R7 and R9.

# Optional Excercise

## 6 Social website

We operate a social web company. Due to its recent rising popularity, response times have increased greatly. The business goal is to have 1500 simultaneous user requests served with less than 4 seconds of response time in average.

a. What minimal throughput should the service infrastructure be designed for, if delays outside our infrastructure (network traffic latency, HTML rendering on the client side) can be estimated as 1 second?

**Solution**

The infrastructure must serve 1500 users with an average of 3 seconds response time. Using Little's law: $N = 1500$, $T = 3\ \frac{\text{s}}{\text{request}}$, so $X = \frac{N}{T} = 500\ \frac{\text{request}}{\text{s}}$

b. According to measurements, an average user request in the redesigned web site takes 20 ms CPU time on the web server, and occupies the database server for 12,5 ms. Currently we have 15 web servers to handle the requests, while the database is replicated to 5 machines. Assuming linear scalability, how much additional units of each kind of server should we buy to meet the above goal?

**Solution**

$T_{\text{CPU}} = 20$ ms $= 0{,}02$ s, $T_{\text{DB}} = 12{,}5$ ms $= 0{,}0125$ s. The CPU and the database must be able to serve at least 500 requests per second in order for the system to do the same (using either sequential or parallel composition). Currently for one instance of the resources: $X^{\max}_{\text{CPU}} = \frac{1}{T_{\text{CPU}}} = 50\ \frac{\text{request}}{\text{s}}$, $X^{\max}_{\text{DB}} = \frac{1}{T_{\text{DB}}} = 80\ \frac{\text{request}}{\text{s}}$. The total maximum throughput of 15 webservers is $750\ \frac{\text{request}}{\text{s}}$, while the for the 5 database server it is only $400\ \frac{\text{request}}{\text{s}}$. So we *need 2 more database servers* in order to reach the desired maximum throughput. (One new database server would increase the maximum throughput to $480\ \frac{\text{request}}{\text{s}}$ only, which is less than the required $500\ \frac{\text{request}}{\text{s}}$.)

c. $(*)$ Calculate the utilization of each kind of server in the extended system. If the goal is to push the average utilization of the servers below 50% even during peak hours, do we need to scale out further?

**Solution**

The maximum throughput of 15 webservers is $X^{\max}_{\text{web}} = 750\ \frac{\text{request}}{\text{s}}$, the required throughput during peak hours is $X_{\text{web}} = 500\ \frac{\text{request}}{\text{s}}$. So their utilizations are $U_{\text{web}} = \frac{X_{\text{web}}}{X^{\max}_{\text{web}}} = \frac{2}{3}$. With the same method: $U_{\text{DB}} = \frac{X_{\text{DB}}}{X^{\max}_{\text{DB}}} = \frac{500}{560} = 0{,}89$.

If we would like to have a 50% utilization ($U = \frac{X_{\text{web}}}{X^{\max}_{\text{web}}}$), then $50\% = \frac{500\ \frac{\text{request}}{\text{s}}}{X^{\max}_{\text{web}}}$ gives $X^{\max}_{\text{web}} = \frac{500\ \frac{\text{request}}{\text{s}}}{0.5} = 1000\frac{\text{request}}{\text{s}}$. For this we need 20 webservers and 13 database servers.

d. Let's consider only 2 webservers and 3 database servers. Create state-based models about the resources in the infrastructure that model the availability of the resources (available or in use). What design decisions do we face? What are the pros and cons of the choices?

**Solution**

Design choices:

- We model the resources *aggregated by their types* based on how many of a certain resource type is in use. This way we will have a 0–1–2 state chain for the webservers and a 0–1–2–3 state chain for the database servers (with the appropriate state transitions between the states). We can get the complete model of the resource pool by taking the asynchronous product of the two state machines.

  The *advantage* of this solution is that it is really simple. We can also easily model the resource allocation of tasks: if the state machine of the required resource is not in its last state (which means none is available), then the allocation is successful and both the task's state machine (got the resource) and the resource's state machine (one less resource instance is available) change states (synchronization). Releasing the allocated resources is done in the same manner.

  The *disadvantage* of this solution is that it doesn't contain any information about the availability of the individual resource instances (except in the all-free and all-used states). Because of this we can't calculate exact utilization values for the individual resource instances, we can only calculate an average value that describes every resource instance.

- We model *every resource instance individually* with a free/taken state pair (or in even more detail). So we will have as many state machines as resource instances in the system. We can get the complete model of the resource pool by taking the asynchronous product of these state machines.
  The *advantage* of this solution is that for example we can calculate some metrics (like utilization) for the individual resource instances. Or something more interesting: we can model the failure and repair process of the individual resource instances in order to analyse the effects of resource failures on system level metrics. The properties of the failure and repair events (like the rate of the event) can be different for every instance, so we are able to model a heterogeneous resource pool (for example webservers of multiple brand) or the degradation of resource instances.
  The *disadvantage* of this solution is that the consumers of resources see more than one resource instance, so it's harder to model resource allocation. In order to allocate a resource instance we have to find a free one first, and after using it we have to free exactly that same one. This process is even more complicated if we need more than one type/instance of resource to perform a task (possibility of deadlocks and starvation). In this case it is preferable (and a used practice) to introduce a resource management component that hides these complications from the consumers.