Mock Exam

# iSAQB® Certified Professional for Software Architecture – Foundation Level (CPSA-F)®

Answer Sheet 2021.2-rev3-EN-20210521

# Explanatory notes on the Mock Exam Certified Professional for Software Architecture – Foundation Level (CPSA-F®)

Explanations to the mock exam Certified Professional for Software Architecture - Foundation Level (CPSA-F®) This examination is a mock exam, which is based on the certification exam of the Certified Professional for Software Architecture - Foundation Level (CPSA-F®) in form and scope. It serves to illustrate the real iSAQB® CPSA® examination as well as to prepare for the corresponding exam. The mock exam consists of 39 multiple-choice questions, which can be evaluated with 1 or 2 points depending on the level of difficulty. At least 60 percent must be achieved to pass the exam.

50.0 points can be achieved in this mock examination, you would need 30.0 points to pass.

The following general rules apply:

- Depending on the level of difficulty and the length of the question, you can achieve a score of 1 or 2 points.
- Correct answers result in plus points, incorrect answers result in a deduction of points, but only with regard to the respective question. If the wrong answer to a question leads to a negative score, this question is evaluated with a total of 0 points.

The multiple-choice questions of the mock exam are divided into three types of questions:

**A-Questions (Single Choice, Single Correct Answer):** Select the only correct answer to a question from the list of possible answers. There is only one correct answer. You receive the specified score for selecting the correct answer.

**P-Questions (Pick-from-many, Pick Multiple):** Select the number of correct answers given in the text from the list of possible answers to a question. Select just as many answers as are required in the introductory text. You receive 1/n of the total points for each correct answer. For each incorrect cross, 1/n of the points are deducted.

**K-Questions (Allocation Questions, Choose Category):** For a question, select the correct of the two options for each answer choice ("correct" or "incorrect" or "applicable" or "not applicable"). You will receive 1/n of the points for each correctly placed cross. Incorrectly placed crosses result in the deduction of 1/n of the points. If NO answer is selected in a line, there are neither points nor deductions.

For a more detailed explanation of the question types and scoring system, further information is available in the CPSA-F examination rules.

The allowed time is 75 minutes for native speakers and 90 minutes for non-native speakers. In order to ensure that the preparation for the exam is as authentic as possible, the processing time should be adhered to and any aids (such as seminar materials, books, internet, etc.) should not be used. The exam can subsequently be evaluated using the solution for this mock exam. Given that the iSAQB® e.V. is indicated as source and copyright holder, the present mock exam may be used in the context of training courses, for exam preparation or it may be passed on free of charge.

However, it is explicitly prohibited to use these exam questions in a real examination.

# Questions

## Question 1

**ID: Q-20-04-01**

| A-Question: | Select one option | 1 point |
|---|---|---|

How many definitions of “software architecture” exist?

| | | |
|---|---|---|
| [ ] | (a) | Exactly one for all kinds of systems. |
| [ ] | (b) | One for every kind of software system (e.g. “embedded”, “real-time”, “decision support”, “web”, “batch”, …). |
| [X] | (c) | A dozen or more different definitions. |

## Question 2

**ID: Q-20-04-02**

| P-Question: | Choose the three best aspects. | 1 point |
|---|---|---|

Which THREE of the following aspects are covered by the term “software architecture”?

| | | |
|---|---|---|
| [X] | (a) | Components |
| [X] | (b) | Cross cutting concepts |
| [X] | (c) | (internal and external) Interfaces |
| [ ] | (d) | Database schema |
| [ ] | (e) | Hardware sizing |

## Question 3

**ID: Q-17-13-01**

| P-Question: | Select the **four** best fitting answers | 2 points |
|---|---|---|

Which FOUR of the following statements about (crosscutting) concepts are most appropriate?

| | | |
|---|---|---|
| [ ] | (a) | Uniform usage of concepts reduces coupling between building blocks. |
| [ ] | (b) | The definition of appropriate concepts ensures the pattern compliance of the architecture. |
| [X] | (c) | Uniform exception handling can be achieved when architects agree with developers upon a suitable concept prior to implementation. |
| [ ] | (d) | For each quality goal there should be an explicitly documented concept. Concepts are a means to increase consistency. |
| [X] | (e) | Concepts are a means to increase consistency. |
| [X] | (f) | A concept can define constraints for the implementation of many building blocks. |
| [X] | (g) | A concept might be implemented by a single building block. |

## Question 4

**ID: Q-17-13-02**

| K-Question: | Select "appropriate“ or „not appropriate" for every line. | 2 points |
|---|---|---|

In your project, three architects and seven developers are working on the documentation of the software architecture. Which methods are appropriate in order to achieve a consistent and adequate documentation, and which are not?

| Appropriate | Not appropriate | | |
|---|---|---|---|
| [X] | [ ] | (a) | The lead architect coordinates the creation of the documentation. |
| [X] | [ ] | (b) | Identical templates are used for the documentation. |
| [ ] | [X] | (c) | All parts of the documentation are automatically extracted from the source code. |

Things like *reasoning* or *alternatives* won't be contained in code, but need to be included in documentation, therefore not **all** parts of documentation can be extracted from source code.

## Question 5

**ID: Q-17-13-03**

| P-Question: | Select the **four** best fitting answers | 1 Punkt |
|---|---|---|

Which FOUR of the following techniques are best suited to illustrate the workflow or behavior of the system at runtime?

| | | |
|---|---|---|
| [X] | (a) | Flowcharts |
| [X] | (b) | Activity Diagrams |
| [ ] | (c) | Depiction of screen flows (sequence of user interactions) |
| [X] | (d) | Sequence diagram |
| [ ] | (e) | Linear Venn diagram |
| [X] | (f) | Numbered list of sequential steps |
| [ ] | (g) | Tabular description of interfaces |
| [ ] | (h) | Class diagrams |

## Question 6

**ID: Q-17-13-04**

| P-Question: | Select the **three** best fitting answers | 1 Punkt |
|---|---|---|

Which THREE of the following principles apply to testing?

| | | |
|---|---|---|
| [X] | (a) | In general, it is not possible to discover all errors in the system. |
| [X] | (b) | In components with many known previous errors, the chances for additional errors are high. |
| [ ] | (c) | Sufficient testing can show that a program is free of errors. |
| [X] | (d) | Testing shows the existence of errors rather than the absence of errors. |
| [ ] | (e) | Functional programming does not allow automated testing. |

## Question 7

**ID: Q-17-03-05**

| K-Question: | Select "True" or „False" for every line. | 1 point |
|---|---|---|

Which of the following statements regarding the information hiding principle are true and which are false?

| True | False | | |
|---|---|---|---|
| [X] | [ ] | (a) | Adhering to the information hiding principle increases flexibility for modifications. |
| [X] | [ ] | (b) | Information hiding involves deliberately hiding information from callers or consumers of the building block. |
| [ ] | [X] | (c) | Information hiding makes it harder to work bottum-up. |
| [ ] | [X] | (d) | Information hiding is a derivative of the approach of incremental refinement along the control flow. |

## Question 8

**ID: Q-20-04-03**

| P-Question: | Choose the **two** best options | 1 point |
|---|---|---|

What are the TWO most important goals of software architecture?

| | | |
|---|---|---|
| [ ] | (a) | Improve accuracy of patterns in structure and implementation. |
| [X] | (b) | Achieve quality requirements in a comprehensible way. |
| [ ] | (c) | Enable cost-effective integration and acceptance tests of the system. |
| [X] | (d) | Enable a basic understanding of structures and concepts for the development team and other stakeholders. |

## Question 9

**ID: Q-20-04-12**

| K-Question: | Select „True" or „False" for every line. | 1 point |
|---|---|---|

Put yourself in the position of a software architect for a large, distributed business application in the banking or insurance domain. Which of the following statements is true and which is false?

| true | false | | |
|---|---|---|---|
| [X] | [ ] | (a) | The architect collaborates with the stakeholders to determine where the requirements and constraints will change often (e.g., business processes, technologies), and designs the architecture such that changes can occur without requiring extensive restructuring of the software architecture. |
| [X] | [ ] | (b) | Required product qualities should drive your architectural decisions. |
| [ ] | [X] | (c) | The software architecture can be designed completely independent of the hardware and infrastructure. |

## Question 10

**ID: Q-20-04-03**

| P-Question: | Choose the **three** best options | 2 points |
|---|---|---|

What are your THREE most important responsibilities as a software architect with respect to requirements?

| | | |
|---|---|---|
| [X] | (a) | Support the business people to specify explicit and concrete quality requirements. |
| [X] | (b) | Help to identify new business opportunities based on your technology know-how. |
| [ ] | (c) | Reject business requirements that contain technical risks. |
| [ ] | (d) | Capture all business requirements in a terminology that can be understood by your development team. |
| [X] | (e) | Check requirements for technological feasibility. |

**Explanation**: Concerning option (c): It's **not** our task to *reject* requirements just because they contain risks. We should identify and communicate those risks, but not reject such requirements.

## Question 11

**ID: Q-20-04-07**

| P-Question: | Choose the **three** best options | 2 points |
|---|---|---|

You are responsible as an architect for keeping a legacy system up and running according to the ongoing requirements of your business. What are the THREE most important action items on your agenda?

| | | |
|---|---|---|
| [ ] | (a) | Negotiating the maintenance budget for your team |
| [X] | (b) | Assuring up-to-date documentation of the deployed system |
| [X] | (c) | Analyzing the impact of new requirements on the current system |
| [ ] | (d) | Encouraging the team members to learn new programming languages |
| [X] | (e) | Suggesting technology updates in addition to the business requirements to your management |

## Question 12

**ID: Q-21-05-01**

| K-Question: | Select „true“ or „false" for every option. | 1 point |
|---|---|---|

Which of the following statements regarding architecture decisions are true, which are false?

| True | False | | |
|---|---|---|---|
| [ ] | [X] | (a) | Architecture decisions never need to be written down because they are already known to the development team. |
| [X] | [ ] | (b) | An architecture decision record helps to make the decision’s context understood. |
| [ ] | [X] | (c) | Once a decision has been made on a central or fundamental framework (e.g. persistence framework), that decision must not be changed. |
| [X] | [ ] | (d) | Quality requirements help significantly with architecture decisions. |

## Question 13

**ID: Q-20-04-09**

| K-Question: | Select „true“ or „false" for every line. | 2 points |
|---|---|---|

Decide for each of the following statements whether it is true or false.

| appropriate | not appropriate | | |
|---|---|---|---|
| [X] | [ ] | (a) | Each iteration of an agile development approach could have a impact on the fundamental architecture decisions. |
| [ ] | [X] | (b) | The total effort spent on architectural work is much higher in iterative projects compared to waterfall projects. |
| [ ] | [X] | (c) | Agile projects do not need architecture documents since the development team uses daily standup-meetings to communicate decisions. |
| [ ] | [X] | (d) | If your systems consist of a set of microservices there is no need for a central architecture document since each service is free to choose its technologies. |

## Question 14

**ID: Q-20-04-10**

| K-Question: | Select „true“ or „false" for every line. | 2 points |
|---|---|---|

Which of the following statements regarding project goals and architectural goals is true and which is false.

| true | false | | |
|---|---|---|---|
| [X] | [ ] | (a) | Project Goals can include functional requirements as well as quality requirements. |
| [X] | [ ] | (b) | Architectural goals are a derived from the quality requirements for the system or product. |
| [ ] | [X] | (c) | Business stakeholders should concentrate on business goals and not interfere with architectural goals. |
| [ ] | [X] | (d) | To avoid conflicts business goals and architectural goals should be non- overlapping sets. |

Explanation:

Business stakeholder might very well have goals like performance, flexibility or security, which are considered "architecture goals".

## Question 15

**ID: Q-20-04-11**

| P-Question: | Select the **two** best fitting answers | 1 point |
|---|---|---|

What does the rule „explicit, not implicit“ mean for architecture work? Choose the TWO best-fitting answers.

| | | |
|---|---|---|
| [ ] | (a) | Architects should avoid recursive structures and replace them by explicit loops. |
| [X] | (b) | Architects should make the assumptions leading to decisions explicit. |
| [ ] | (c) | Architects should explicitly insist on natural language explanations (i.e. comments) for each building block. |
| [ ] | (d) | Architects should explicitly insist on written or at least verbal justifications for development effort estimates from their team. |
| [X] | (e) | Architects should make prerequisites for their decisions explicit. |

## Question 16

**ID: Q-20-04-19**

| P-Question: | Select the **three** best fitting answers | 1 point |
|---|---|---|

Identify the THREE most appropriate examples for typical categories of software systems.

| | | |
|---|---|---|
| [X] | (a) | Batch system |
| [X] | (b) | Interactive online system |
| [ ] | (c) | Linnés system. |
| [X] | (d) | Embedded real-time system. |
| [ ] | (e) | Integration test system. |

## Question 17

**ID: Q-20-04-32**

| P-Question: | Select the **three** best fitting answers | 1 point |
|---|---|---|

There are many approaches that lead to a software architecture. Which of the following are the THREE most often found in practice?

| | | |
|---|---|---|
| [ ] | (a) | User interface driven design |
| [X] | (b) | Domain driven design |
| [X] | (c) | View based architecture development |
| [X] | (d) | Bottom-up design |
| [ ] | (e) | Majority voting |

## Question 18

**ID: Q-20-04-38**

| P-Question: | Select the **three** most often used architecture views | 1 point |
|---|---|---|

Several architecture development methods suggest a view-based approach. Which three of the following views are most often used?

| | | |
|---|---|---|
| [ ] | (a) | Physical database view |
| [X] | (b) | Context view |
| [X] | (c) | Building Block/Component view |
| [ ] | (d) | Test-driven view |
| [ ] | (e) | Configuration view |
| [X] | (f) | Runtime view |

## Question 19

**ID: Q-20-04-22**

| P-Question: | Select the **two** best fitting answers | 1 point |
|---|---|---|

When documenting a building block of your software architecture, which two information should the black-box description contain?

| | | |
|---|---|---|
| [X] | (a) | Public interfaces. |
| [X] | (b) | Responsibility of the building block. |
| [ ] | (c) | Internal structure of the building block. |
| [ ] | (d) | Specification of the implementation details. |

## Question 20

**ID: Q-20-04-17**

| P-Question: | Select the **two** best fitting answers | 1 point |
|---|---|---|

Which prerequisites have to be fulfilled before developing a software architecture? Pick the TWO most appropriate answers.

| | | |
|---|---|---|
| [ ] | (a) | The requirements specification for the system is complete, detailed and consistent. |
| [X] | (b) | The most important qualities for the system are known. |
| [X] | (c) | Organizational constraints are known. |
| [ ] | (d) | The programming language has been selected. |
| [ ] | (e) | Hardware for the development team is available. |

In most cases it is unrealistic to have *complete* requirements specification. Often it is enough to have an overview and know certain details (e.g. quality requirements).

## Question 21

**ID: Q-20-04-18**

| P-Question: | Select the **three** best fitting answers | 1 point |
|---|---|---|

Which factors can influence the design of a software architecture? Pick the THREE most appropriate answers.

| | | |
|---|---|---|
| [X] | (a) | Political. |
| [X] | (b) | Organizational. |
| [X] | (c) | Technical. |
| [ ] | (d) | Virtual. |

## Question 22

**ID: Q-20-04-18**

| A-Question: | Select one option | 1 Point |
|---|---|---|

Which of the following qualities can most likely be improved by using a layered architecture?

| | | |
|---|---|---|
| [ ] | (a) | Runtime efficiency (performance). |
| [X] | (b) | Flexibility in modifying or changing the system. |
| [ ] | (c) | Flexibility at runtime (configurability). |
| [ ] | (c) | Non-repudiability. |

## Question 23

**ID: Q-20-04-33**

| A-Question: | Select one option | 1 Point |
|---|---|---|

For which kind of system can the Blackboard Architecture pattern be used?

[ ] (a) Hard real-time systems

[X] (b) Rule-based systems

[ ] (c) Linnés systems

[ ] (c) Safety critical systems

## Question 24

**ID: Q-20-04-20**

| A-Question: | Select one option | 1 Point |
|---|---|---|

Which goals are you trying to achieve with the dependency inversion principle?

[ ] (a) Big building blocks shall not depend on small building blocks.

[ ] (b) Components shall be able to create dependent components more easily.

[X] (c) Building blocks shall only depend on each other via abstractions.

## Question 25

**ID: Q-20-04-21**

| K-Question: | Select „tight coupling“ or „loose coupling" for each line. | 1 point |
|---|---|---|

What are characteristics of tight (high) or loose (low) coupling?

| tight coupling | loose coupling | | |
|---|---|---|---|
| [X] | [ ] | (a) | Building blocks directly call dependent building blocks, i.e., without using indirect calls via interfaces or abstractions. |
| [X] | [ ] | (b) | Building blocks use shared complex data structures. |
| [X] | [ ] | (c) | Building blocks use a shared table (for read- and write operations) within a relational database. |
| [ ] | [X] | (d) | When designing building blocks, you have consistently applied the dependency inversion principle. |

## Question 26

**ID: Q-20-04-14**

| P-Question: | Select the **two** best fitting answers | 2 points |
|---|---|---|

Which two statements about the principle „Don’t repeat yourself” (DRY) fit best? In other words: What could happen, if parts of the source code or configuration do exist in multiple copies in the system?

| | | |
|---|---|---|
| [ ] | (a) | DRY reduces security. |
| [X] | (b) | Strict adherence to DRY could lead to higher coupling. |
| [X] | (c) | The components of the system that contain redundant code can be improved independently of each other. |
| [ ] | (d) | Adherence to DRY leads to additional attack vectors in IT security. |
| [ ] | (e) | Applying the Layer patterns allows a consistent application of the DRY principle. |

## Question 27

**ID: Q-20-04-15**

| K-Question: | Select „true“ or „false" for every line. | 2 points |
|---|---|---|

You can communicate aspects of your software architecture verbally and/or in writing. How do these variants correlate? Decide for each of the following statements whether it is true or false.

| true | false | | |
|---|---|---|---|
| [X] | [ ] | (a) | Verbal communication should supplement written documentation. |
| [ ] | [X] | (b) | Feedback to architecture decisions should always be done in writing to ensure traceability. |
| [ ] | [X] | (c) | Written documentation should always precede verbal communication. |
| [ ] | [X] | (d) | Architects should pick one variant (verbal or written) and stick to this choice during the whole development. |

- Sometimes verbal communication needs to come first, there is no general rule.
- Feedback should not be restricted to written statements.

## Question 28

**ID: Q-20-04-37**

| K-Question: | Select „true“ or „false" for every line. | 2 points |
|---|---|---|

Which of the following statements about notations for architectural views is true and which is false?

| true | false | | |
|---|---|---|---|
| [ ] | [X] | (a) | Business Process Model & Notation (BPMN) should only be used by Business Analysts and not for architecture documentation. |
| [ ] | [X] | (b) | UML deployment models are the only way to document the mapping of software components to infrastructure. |
| [X] | [ ] | (c) | UML Package Diagrams can be used to capture the building-block view of software architectures. |
| [X] | [ ] | (c) | As long as the notation is explained (e.g. by a legend), any notation can be sufficient to describe building block structures and collaboration. |

## Question 29

**ID: Q-20-04-13**

| P-Question: | Select the **two** best fitting answers | 1 point |
|---|---|---|

Which architectural views have the most practical application for developing software architectures? 1 point

| | | |
|---|---|---|
| [ ] | (a) | Pattern View. |
| [ ] | (b) | Observer View. |
| [X] | (c) | Building-Block View (Component View). |
| [X] | (d) | Deployment View. |

## Question 30

**ID: Q-20-04-23**

| P-Question: | Select the **two** most appropriate answers | 1 point |
|---|---|---|

The context view might contain a business context and a technical context, or both. Pick the two most appropriate answers that apply to the technical context.

| | | |
|---|---|---|
| [X] | (a) | The technical context contains the physical channels between your system and its environment. |
| [ ] | (b) | The technical context contains all the infrastructure on which the components of your system are deployed. |
| [ ] | (c) | The technical context should include hardware pricing or pricing of cloud services used as infrastructure for your architecture. |
| [ ] | (d) | The technical context contains information about the chosen programming language as well as all frameworks used to implement your software architecture. |
| [X] | (e) | The technical context might contain different elements than the business context. |

## Question 31

**ID: Q-20-04-24**

| P-Question: | Select the **two** best reasons | 1 point |
|---|---|---|

Software architecture documentation could contain descriptions of cross-cutting concerns. Pick the TWO best reasons why documentation of cross-cutting concerns is useful.

| | | |
|---|---|---|
| [ ] | (a) | Cross-cutting concepts should focus on the domain and be free of technical information. |
| [X] | (b) | Aspects or concepts that are used in multiple parts of your software architecture should be described in a non-redundant way. |
| [X] | (c) | Cross-cutting concepts can be reused in more products within the same organization. |
| [ ] | (d) | Cross-cutting concepts should be implemented by specialists. Therefore, separate documentation is useful. |

## Question 32

**ID: Q-20-04-25**

| K-Question: | Select „true“ or „false" for every line. | 2 points |
|---|---|---|

What are guidelines for good interface design? Check which of the following statements are true and which are false.

| true | false | | |
|---|---|---|---|
| [X] | [ ] | (a) | Use of interfaces should be easy to learn. |
| [X] | [ ] | (b) | The client code should be reasonably easy to understand in relation to the functional complexity. |
| [ ] | [X] | (c) | An interface should provide access to a comprehensive set of implementation details. |
| [X] | [ ] | (d) | Interface specifications should contain functional and non-functional aspects. |
| [ ] | [X] | (e) | Local and remote calls to this interface should behave identically in all aspects. |

**Explanation**

Regarding option (e), "identical behavior in all aspects": It’s technically not feasible to have *identical* behavior, at least concerning latency, and response time.

A more detailed explanation can be found in the (rather famous) Fallacies_of_distributed_computing

## Question 33

**ID: Q-20-04-26**

| K-Question: | Select „true“ or „false" for every line. | 1 point |
|---|---|---|

One definition says: “Software architecture is the sum of all the decisions you have taken during development. Check which of the following statements about architectural/design decision is true and which is false.

| true | false | | |
|---|---|---|---|
| [X] | [ ] | (a) | Architectural decisions can impact the structure of the building block or components. |
| [ ] | [X] | (b) | Software architects shall justify all design decisions in writing. |
| [X] | [ ] | (c) | Architectural decisions can have interdependencies between each other. |
| [X] | [ ] | (d) | Tradeoffs between conflicting quality requirements should be explicit decisions. |

Not *all* decisions need to be justified in writing - as the requirement for *written* documentation depends on the situation, the team, the system and other factors.

## Question 34

**ID: Q-20-04-31**

| K-Question: | Select „typical“ or „not typical" for every line. | 2 points |
|---|---|---|

Which of the following statements are typical reasons for maintaining adequate architecture documentation and which are not typical reasons?

| typical | not typical | | |
|---|---|---|---|
| [X] | [ ] | (a) | To support onboarding of new developers. |
| [ ] | [X] | (b) | To support the automated testing approach of the system. |
| [X] | [ ] | (c) | To support the work of distributed teams. |
| [X] | [ ] | (d) | To assist in future enhancements of the product. |
| [X] | [ ] | (e) | To conform to regulatory or legal constraints. |

| | | | |
|---|---|---|---|
| [ ] | [X] | (f) | To ensure that developers have enough work to do. |

## Question 35

**ID: Q-20-04-30**

| K-Question: | Select „conflicting“ or „not conflicting" for every line. | 1 point |
|---|---|---|

Which of the following pairs of qualities are usually in conflict to each other, and which are not?

| conflict | no conflict | | |
|---|---|---|---|
| [ ] | [X] | (a) | Understandability – Readability. |
| [X] | [ ] | (b) | Usability – Security. |
| [X] | [ ] | (c) | Runtime configurability – Robustness. |
| [ ] | [X] | (d) | Security – Legal Compliance. |

## Question 36

**ID: Q-20-04-27**

| P-Question: | Select the **two** best alternatives | 1 point |
|---|---|---|

ISO 25010 provides generic quality characteristics for software systems. How can quality requirements concerning these characteristics be made more concrete? Pick the two best alternatives.

| | | |
|---|---|---|
| [ ] | (a) | By developing UI prototypes. |
| [ ] | (b) | By defining explicit interfaces. |
| [X] | (c) | By discussing or writing scenarios. |
| [ ] | (d) | By creating automated tests. |
| [X] | (e) | By creating a quality tree. |

## Question 37

**ID: Q-20-04-28**

| P-Question: | Select the **four** best alternatives | 1 point |
|---|---|---|

Which of the following alternatives are most suitable for supporting a qualitative analysis of your software architecture? Pick the four best alternatives.

| | | |
|---|---|---|
| [X] | (a) | Quantitative dependency analysis. |
| [X] | (b) | Architecture models. |
| [X] | (c) | Quality scenarios. |
| [ ] | (d) | Team size. |
| [X] | (e) | Log files. |
| [ ] | (f) | Organizational structure. |

## Question 38

**ID: Q-20-04-29**

| P-Question: | Select the **two** best fitting answers | 2 points |
|---|---|---|

You try to analyze your architecture quantitatively. Which are the two most appropriate indicators for architectural problem areas?

| | | |
|---|---|---|
| [X] | (a) | High coupling of components. |
| [ ] | (b) | Names of public methods do not reflect their purpose. |
| [ ] | (c) | Missing comments. |
| [X] | (d) | Clusters of errors in certain building blocks of the system. |
| [ ] | (e) | Number of test cases per component. |

## Question 39

**ID: Q-20-04-36**

| P-Question: | Select the **three** best fitting answers | 1 point |
|---|---|---|

You try to quantitatively analyze your architecture. Which three of the following properties can you measure reliably in your software architecture? Pick the three best fitting answers.

| | | |
|---|---|---|
| [X] | (a) | Size of building blocks (e.g. LOC). |
| [X] | (b) | Change rate of the source code of components. |
| [ ] | (c) | Cohesion of the architectural components. |
| [ ] | (d) | Security level of a component. |
| [X] | (e) | Number of the developers that contributed to a specific component. |

### Explanation

- Size can easily and reliably be measured when statically analyzing source code (lines-of-code metric is a reliable size metric)
- change-rate and number-of-developers-per-component can reliably be measured when taking the version control history into account, which is perfectly feasibly with systems like git, subversion or similar tools that are widely used in development.