What is advanced in Software Engineering?

Dependable software (that always works) which has an extended lifecycle (runs decades, not years) and is large, complex and integrated.

Describe the main ideas behind the DevOps movement!

- The DevOps movement is a philosophical idea where the traditional boundaries between development and operations are removed.
- It is more about breaking down walls than classic "who does what".
- It is not about who does something, it is about how it is done.
- It is about considering operations from the beginning.
- It is about knowing how the other side works.
- It is about facilitating communication.

What is meant by technical debts? Name three examples of technical debts and their possible causes!

Technical debts are created when a developer opts for a faster solution instead of a more sustainable, readable, maintainable one. Examples:

- no documentation of the architecture. Possible cause: the customer didn't want to pay for it, so no documentation has been created
- bad modularization or architecture. Possible cause: the development team didn't have enough skills or experience when they created the software
- a lot of TODOs in the code base. Possible cause: the developers were under time pressure and decided to implement a "quick & dirty" solution and refactor it later

What are the differences between localization and internationalization? Explain typical focus points!

- Internationalization (I18n): The preparation of a (software) product for use in the global market (language, culture), usually done only once (no source code changes necessary).
- Localization (L10n): Performing specific adaptations necessary to launch a product in a specific locale (done for every locale)

Focus points:

Language & Text

- Character encoding (UTF-8 should be de facto standard)
- Orientation: Left to right vs. right to left vs. vertical
- Sorting
- Pluralization

Culture

- Names and titles
- Weights and measurements, paper sizes

- Telephone, Addresses, Postal codes

Conventions

- Currency format
- Date, Time, Time zones and Calendars
- Number format

Compare components and services after Fowler

- A component is a glob of software, that is used without change by an application, that has no control over the writer of the component. (used locally e.g. jar)
- A service is also used by a foreign application, but it is accessed remotely (e.g. web service)

Name 5 CI Principles after Fowler

1. Maintain a code repository
2. Automate the build
3. Make the build self-testing
4. Everyone commits to the baseline every day
5. Every commit (to baseline) should be built
6. Keep the build fast
7. Test in a clone of the production environment
8. Make it easy to get the latest deliverables
9. Everyone can see the results of the latest build
10. Automate deployment

What is the key concept of microservices? Name and explain two typical challenges when developing microservices!

Microservices are an architecture pattern, where the software is split into small, independently deployable, modular services, where each service runs in a unique process and has exactly one goal. They communicate with each other for example over a message bus.
Challenges:

- using many different technologies ("best tool for the job" philosophy can be hard to maintain for a company.
- because every service is deployed on its own versioning is complicated
- debugging is harder because it involves many services

When to use which form of transaction management?

- If running inside a suitable container, declarative transactions are usually the best way (Container Managed Transactions (CMT))
- Local transaction models are used when the database should handle the transaction (auto-commit)
- Programmatic transaction models should be used when the developer wants to specify the start and end of a transaction (no auto-commit)

Compare logging and auditing. Explain two characteristics of good logging output.

- Logging: Technical, text based output primarily used for detecting and debugging problems (stored for weeks or month). Helps to understand the how the system works / in which state the system is.
- Auditing: Domain specific, fine grained and structured output for tracing user activity (stored for years 10+ / 30+, compliance reasons). Helps to see what actions where perform by the users.

Good log output:

- Only use limited log levels and use them with clear rules (DEBUG, INFO, WARNING, ERROR)
- Always provide a reference (e.g. userId or sessionId) and log information about the context

Explain the idea behind Continuous Integration! What's the difference between Continuous Integration, Continuous Delivery and Continuous Deployment?

- Continuous Integration is the process of automating the build process. Work is constantly merged to the baseline and built and tested automatically there. (Execute a full build after every commit)
- Continuous Delivery additionally delivers code to a staging environment and runs acceptance tests there.
- Continuos Deployment means automatic deployment of code from SCM to production. Needs Continuos Integration and Continuos Delivery to be in place. Automatically runs smoke tests.

What are cross cutting concerns? Why should cross cutting concerns be separated from business logic?

- Cross-cutting concerns are code parts which are reused a lot within the software (e.g. logging, transactions). They span over multiple points of the application.
- They should be separated from business logic (e.g. using aspect-oriented programming) in order to prevent code duplication and to keep the business code shorter and more readable.

Explain the differences between Build Time Modularization and Runtime Modularization and name one popular Java technology for one of them!

- Build time modularization contains of a single bundle (container). Updates requires complete redeployment. (e.g. Maven)
- Runtime modularization: multiple bundles with multiple jars. Updates can be done by partially redeploying. (e.g. OSGI)

What is Dependency Management?

Dependency management is used to manage external components and libraries which are used in the software.

- Declare which libraries you are using
- Declare which version of a library you are using
- Declare in which context you are using the library (test vs. production)
- Declare where these libraries are coming from
- Have these libraries declare, which libraries they are using
- Automatically retrieve all required libraries from a/your repository

What problems can be solved with Dependency Management?

- Loose version information
- No standardized naming (convention)
- Loose trace to source (e.g. download page)
- No information about transitive dependencies
- Difficult to manually find updated versions
- SCM (e.g. Git) not built for versioning binaries (no diff, bad handling binary files, high resource usage)

Name two Dependency Management Tools!

- Maven: Manage tools in one: manage dependencies, build projects, run tests, release, execute (custom) plugins
- Gradle: DSL instead of XML, Official Android build tool, more flexible scripting

Explain dependency injection!

Dependency Injection is a pattern, where the gluing of objects is separated from the implementation. Therefore all implementation is against the API. There is a central definition and container that creates and binds objects together.

Name possibilities and advantages of dependency injection!

- DI supports code reuse
- It’s a very useful technique for testing, since it allows dependencies to be mocked or stubbed out (independently testing classes)
- DI supports different bindings for different environments
- DI supports lazy creation of objects

What is the big benefit if you hit a bug in FOSS?

You don't need workarounds in your own code, instead you can fix bugs directly in the OSS libraries you use.

Name 5 FOSS (Free and Open Source Software) business models!

- Adding commercially available value on top of a base OSS offering
- Professional Training of your OSS software
- Embedding OSS into hardware (e.g. most routers use Linux, iptables, Android)
- Service Contracts (i.e. 'insurance' for your project)
- Project Consulting (i.e. Help other companies to save money by using OSS)

Name 3 things to avoid while creating FOSS business models!

- Don't try to sell the same product you give away for free
- Respect the freedom! (i.e. Don't force/restrict others to do something only because you need it). Respect the community
- Don't push a project into one corner just because a customer likes it that way
- Be aware that OSS project planing is different than company projects

Explain configuration management. Name three main types of CM!

Configuration Management is a management process for establishing and maintaining consistency of a product's performance, its functional and physical attributes, with its requirements, design and operational information, throughout its life.

- Build configuration
- Product configuration
- Application server/database configuration
- OS configuration
- System configuration

Name and explain reasons for software aging!

- Lack of movement: Failure to modify product to meet changing needs.
- Ignorant surgery: Result of the changes that are made.

What problems does software aging create and what are preventive measures?

Problems

- Inability to keep up (growth)
- Reduce of performance over time because of poor design
- Decreasing reliability because of error injection

Preventive measures

- Design and plan for change
- Documentation and Reviews
- Restructuring including partial replacement (amputation)
- Plan for retirement and replacement

Name attributes of dependability

- Maintainability (can be repaired and modified)
- Availability (always ready)
- Reliability (always correct)
- Confidentiality (no unauthorized disclosure of information)
- Integrity (no invalid system states)
- Safety (no catastrophic consequences for the user)

What are two means against threats of dependability, explain them!

- Fault forecasting: Qualitative (identify, classify, rank) and Quantitative (probability model)
- Fault prevention: By quality control and software design (structured programming, information hiding, modularization)
- Fault tolerance: By error detection and recovery, error handling (rollback/rollforward), redundancy and fault isolation
- Fault removal: By static/dynamic verification, diagnosis and correction
- Fault injection: To test error handling and corrective/preventive maintenance

What are Collective Intelligence Systems (CIS)?

Harness the collective intelligence of connected groups of people by providing a web-based environment to share, distribute and retrieve topic-specific information

Name 3 risk factors for collective intelligence!

- If CIS is not well integrated in users' workflow routine, the CIS will not be used
- Focusing more on the software side, and neglecting the user base side
- Cannibalization of user activity by other CIS
- Handling of security and privacy of user data

Advantages and disadvantages of centralized CIS and decentralized CIS

Centralized:

- Pro: Constant quality of service for all participants.
- Pro: Data is stored in one place.
- Pro: More resources for system maintenance, data security, platform evolution.
- Pro: Single point of access.
- Pro: Effective information exchange due to very-large scale user base.
- Con: Single point of failure.
- Con: Prone to censorship and systematic infiltration by government organizations.
- Con: Often closed/proprietary system source code.
- Con: Lots of influence concentrated in a single organization

Decentralized:

- Pro: Multiple points of access
- Pro: Robust: infiltration only of individual points; easy hosting of new nodes.
- Pro: Often system source code is open source (= auditable)
- Pro: Easier to operate on non-profit basis due to lower resource needs of individual nodes.
- Con: Quality of service dependent on individual node.
- Con: Each node is responsible for its maintenance and data security.
- Con: Less effective information exchange due to fragmentation of user base.
- Con: User contributions are stored on an individual node.

Name 5 CI (Collective Intelligence) design patterns and explain their solutions!

- Tagging: Enables users to categorize content on their own using keywords.
- Rating: Enables users to express their like or dislike about the content so that other users can benefit.
- Comments: Enable users to express their own opinion to a topic and encourages discussions with others.
- Hashtags: Enables users to add keywords/tags using # symbol among text to facilitate discoverability later.
- Recommendations: Seek to predict content the user might be interested in based on the other users' behavior and decisions.
- Follow/subscribe: Enable users to define specific content they are interested in and that they would like to monitor.

Name two types of CIS, and for each 3 examples

- Social network services (Facebook, Twitter, LinkedIn, Instagram)
- Media / Content Sharing (YouTube, Soundcloud, Flickr, Slideshare)
- Knowledge Creation (Wikipedia, Stack Overflow, Yelp)

Name 4 agile principles of the agile manifesto!

- Individuals and interactions over processes and tools
- Working software business results over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Name 5 Exception Handling Anti-Patterns!

- Log an exception and throw it: Do either one of them.
- Catch "Exception" (i.e. all exceptions, not one in particular): It's like a fisher's net - you do not know what you will catch.
- Destructive wrapping: Always pass the causing exception.
- Catch and ignore
- Throw from within finally: Will swallow any other exception.

Difference between sampling and tracing in performance profiling?

- Tracing: Usually done through byte code instrumentation. Delivers invocation counts. Can significantly influence runtime performance. Not suitable for production environments.

- Sampling: Periodically queries stacks of running threads to estimate the slowest parts of the code. No invocation counts. Negligible performance impact.

Difference between role based and permission based authorization Benefits of permission based?

- Role-based authorization is simple and easy to grasp but permission based authorization is more flexible.
- Permission-based authorization can be much more fine-grained which can result in bigger administration costs.

Difference between Clustering and Load-Balancing?

Clustering means you run a program on several machines (nodes). One reason why you want to do this is Load balancing. If you have too much load/work to do for a single machine you can use a cluster of machines instead. A load balancer then can distribute the load over the nodes in the cluster.

Name and describe 4 naming conventions, that help maintaining clean code

- If you need a comment you are doing something wrong!
- Descriptive (long) name > short name
- Precise names for small classes > generic names for large classes
- Clarity is king
- Length of a name should correspond to the size of its scope

What is the "open-closed-principle"?

Classes should be open for extension – closed for modification

Explain the difference between authentication and authorization and name two examples for each one!

Authentication verifies the identity of a user. Authorization determines the access rights.

Authentication Methods:

- Username/Password - Easy to implement and use but insecure
- Certificate based (client authentication) - Complex to roll out and manage but high security
- Smart card, Biometric - Needs client side support
- Token based / Single Sign On (SSO) - Delegate Authentication, use central identity (OAuth, SAML)

Authorization Methods:

- Role based access control (RBAC) - For resource based systems. Simple and easy to grasp
- Permission based - Simple action based, Complex expressions. Fine grained and flexible. Well supported by standard technologies
- Access control list (ACL) - Delivers fine grained control on (object) instance level. Complex to maintain
- Rule based - For complicated and frequently changing business requirements