**AS 2024/25**

Inšpirácia na výber projektových tém

* Lean architecture and related
  + James O. Coplien and Gertrud Bjørnvig. Lean Architecture: for Agile Software Development. Wiley, 2010.
    - <https://epdf.pub/lean-architecture-for-agile-software-development.html>
    - <http://flambeau.fastmail.jp/ebooks/LeanArchitecture.pdf> (draft)
  + DCI - Data Context Interaction: A New Role-Based Paradigm for Specifying Collaborating Objects, <http://www.fulloo.info/>
  + Trygve Reenskaug. Working With Objects: The OOram Software Engineering Method. Prentice Hall, 1995. <http://heim.ifi.uio.no/~trygver/1996/book/WorkingWithObjects.pdf>
* Analysis patterns
  + M. Fowler. Analysis Patterns: Reusable Object Models. Addison–Wesley, 1996. <http://uml2.narod.ru/files/docs/13/AnalysisPatterns.pdf>
* People and architecture
  + J. O. Coplien and N. B. Harrison. Organizational Patterns of Agile Software Development. Prentice Hall, 2004. [http://web.archive.org/web/20050514110633/http://easycomp.info.uni-karlsruhe.de/%7Ejcoplien/HarrisonCoplien.pdf](http://web.archive.org/web/20050514110633/http:/easycomp.info.uni-karlsruhe.de/%7Ejcoplien/HarrisonCoplien.pdf)
  + M. Conway. Conway's Law. Mel Conway's Home Page, 2012. <http://www.melconway.com/Home/Conways_Law.html>
  + A. Cockburn. Agile Software Development: The Cooperative Game. 2nd Edition. Addison-Wesley, 2006.
* Essentials and deeper connotations
  + M. Fowler. Making Architecture Matter. OSCON 2015, Portland, USA (video). <https://www.youtube.com/watch?v=DngAZyWMGR0>
  + patternlanguage.com, <https://www.patternlanguage.com/>
    - Computer Science, <https://www.patternlanguage.com/comp-sci/comp-sci.html>
  + C. Alexander. The Timeless Way of Building. Oxford University Press, 1977.
  + C. Alexander et al. A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977.
* Architecture and design patterns
  + Pattern-Oriented Software Architecture (POSA)
    - K. Henney and F. Buschmann. Pattern-Oriented Software Architecture. OOP 2008. Presentation slides. Munich, Germany, 2008. <http://www.dre.vanderbilt.edu/~schmidt/POSA-tutorial.pdf>
    - [POSA1] F. Buschmann, K. Henney, and D. C. Schmidt. Pattern-Oriented Software Architecture, A System of Patterns, Volume 1. John Wiley & Sons, 1996. <https://epdf.pub/pattern-oriented-software-architecture-volume-1-a-system-of-patterns.html>
    - [POSA2] D. Schmidt, M. Stal, H. Rohnert, and F. Buschmann. Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects. Volume 2. John Wiley & Sons, 2000. <http://index-of.co.uk/SE/Pattern-Oriented%20Software%20Architecture,%20Volume%202.pdf>
    - [POSA3] M. Kircher and P. Jain. Pattern-Oriented Software Architecture: Patterns for Resource Management. Volume 3. John Wiley & Sons, 2001. <https://epdf.pub/pattern-oriented-software-architecture-volume-3-patterns-for-resource-management.html>
    - [POSA4] F. Buschmann et al. Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing. Volume 4. John Wiley & Sons, 2007. <https://epdf.pub/pattern-oriented-software-architecture-a-pattern-language-for-distributed-comput58193.html>
    - [POSA5] F. Buschmann, K. Henney, and D. C. Schmidt. Pattern-Oriented Software Architecture: On Patterns and Pattern Languages. Volume 5. John Wiley & Sons, 2007. <https://epdf.pub/pattern-oriented-software-architecture-vol5-on-patterns-and-pattern-languages.html>
  + [GoF] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides.Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995. <https://epdf.pub/design-patterns-elements-of-reusable-object-oriented-software.html>
  + PLoP, <https://www.hillside.net/plop/2019/papers/proceedings/> (just change the year in the address to get to the websites of previous conferences)
  + EuroPLoP, <https://europlop.net/content/proceedings-0>
  + VikingPLoP, <https://vikingplop.org/proceedings/>
  + I. Sommerville, Software Engineering, Addison-Wesley, 2011.  
    <https://dinus.ac.id/repository/docs/ajar/Sommerville-Software-Engineering-10ed.pdf>
  + SourceMaking. Design Patterns. 2020. <https://sourcemaking.com/design_patterns>
* Game engine architecture
  + TrinityCore (WoW private server): <https://www.trinitycore.org/>
    - Source code: <https://github.com/TrinityCore/TrinityCore>
* Machine-learning architecture
  + Mary Shaw. Myths and Mythconceptions: What Does It Mean to Be a Programming Language, Anyhow? Proceedings of the ACM on Programming Languages, 4(HOPL), Article 234, June 2020. <https://doi.org/10.1145/3480947>
    - See the section entitled *The AI Revolution Myth*
* Economics-driven software architecture
* Distributed software architecture
  + G. Hohpe and B. Woolf. Enterprise Integration Patterns: Designing, Building, and Depolying Messaging Solutions. Addison Wesley, 2003. <https://raw.githubusercontent.com/vaquarkhan/vaquarkhan/master/integration%20design%20pattern/Addison%20Wesley%20-%20Enterprise%20Integration%20Patterns%20-%20Designing%2C%20Building%20And%20Deploying%20Messaging%20Solutions%20-%20With%20Notes.pdf>
* Microservices
  + M. Fowler. Microservices, 2014. <https://martinfowler.com/articles/microservices.html>
  + M. Fowler. Microservices, 2019. <https://martinfowler.com/microservices/>
  + <https://microservices.io/patterns/microservices.html>
* Software defined infrastructure
* Architecture and test driven development
  + Jim Coplien and Bob Martin Debate TDD
    - <https://www.youtube.com/watch?v=KtHQGs3zFAM>
    - <https://www.infoq.com/interviews/coplien-martin-tdd/>
  + Robert C. Martin. TDD Harms Architecture. The Clean Code Blog, 2017. <https://blog.cleancoder.com/uncle-bob/2017/03/03/TDD-Harms-Architecture.html>
  + D. S. Janzen and H. Saledian. Does Test-Driven Development Really Improve Software Design Quality? IEEE Software 25(2), 2008. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.304.1723&rep=rep1&type=pdf>
* Software architecture recovery
  + Some software worth exploring:
    - Apache Tomcat, <http://tomcat.apache.org/>
    - Asterisk, <https://www.asterisk.org/>
    - DOSBox, <https://www.dosbox.com/>
    - Eclipse, <https://www.eclipse.org/>
    - Elasticsearch, <https://www.elastic.co/guide/index.html>
    - Firefox, <https://www.mozilla.org/>
    - Gimp, <https://www.gimp.org/>
    - Hadoop, <https://hadoop.apache.org/>
    - Hibernate (ORM), <https://hibernate.org/orm/documentation/5.4/>
    - KODI, <https://kodi.tv/>
    - Microsoft CNTK, <https://docs.microsoft.com/en-us/cognitive-toolkit/>
    - Notepad++, <https://notepad-plus-plus.org/>
    - OpenOffice, <https://www.openoffice.org/sk/>
    - Tensorflow, <https://www.tensorflow.org/guide>
    - Zimbra, <https://www.zimbra.com/>
  + Pattern detection
* Software architecture refactoring
  + M. Fowler et al. Refactoring: Improving the Design of Existing Code. Addison Wesley, 1999.
  + J. Kerievsky. Refactoring to Patterns. Addison Wesley, 2004.
  + Joshua Kerievsky. The Art of Refactoring - Agile Singapore Conference 2016 (video). <https://www.youtube.com/watch?v=sxSc4UqtqQA>
* Software product lines
  + J. Bosch. Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach. Pearson Education, 2000.

Ďalšie zdroje z ktorých je možné čerpať nápady na projektové témy (nezaradené):

* Alessio Gambi: Software Architecture, Software Engineering, Saarland University, 2017
* Layered architecture, <https://www.baeldung.com/cs/layered-architecture#definitions>
  + 5 layers in software architecture, <https://www.indeed.com/career-advice/career-development/what-are-the-layers-in-software-architecture>
* Architectural patterns and styles
  + <https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ee658117(v=pandp.10)?redirectedfrom=MSDN>
  + <https://herbertograca.com/2017/07/28/architectural-styles-vs-architectural-patterns-vs-design-patterns/>
  + <https://www.geeksforgeeks.org/difference-between-architectural-style-architectural-patterns-and-design-patterns/>
* Best Practices for Middleware and Integration Architecture Modernization with Apache Camel, <https://www.youtube.com/watch?v=d1Hr78a7Lww>