Glossary

Acronyms

ABE anology-based effort estimation; see pages 8, 9, 38, 76, 80, 81, 84, 111, 112, 149, 155 and 156

AI artificial intelligence; see pages 7 and 89

API application programming interface; see pages 14, 30, 32, 36, 37, 75, 80, 113, 126, 153, 158 and 162

AUC Area Under Curve; see page 111

CI continuous integration; see page 37

CPU central processing unit; see pages 39, 64–66 and 131

CSS Cascading Style Sheets; see page 100

CSV comma-separated values; see pages 61 and 106

DNN deep neural network; see pages 8, 9, 81, 111, 155 and 156

DoD Definition of Done; see pages 6, 7, 73 and 125

ER entity–relationship diagram; see pages 47–50

FTE full-time equivalent; see pages 49 and 122

GPU graphics processing unit; see pages 37, 38 and 149

HCI human-computer interaction; see page 95

HTML HyperText Markup Language; see pages 100, 106, 128 and 138

HTTPS HyperText Transfer Protocol Secure; see pages 33, 61 and 138

InfoVis information visualization; see page 97

JDBC Java Database Connectivity; see page 59

JSON JavaScript Object Notation; see pages 30, 37, 38, 60, 100, 103, 106 and 153

LDAP Lightweight Directory Access Protocol; see pages 36, 47, 48, 58 and 59

Lin linear regression algorithm; see pages 8, 9, 80, 82 and 86

LSTM Long Short-Term Memory; see pages 91 and 160

MC Monte Carlo simulation; see pages 8, 9, 76, 80, 82, 86, 103 and 135

ML machine learning; see pages 7, 8, 12, 71, 76 and 89

NN neural network; see pages 8, 9, 80, 81 and 149

PB product backlog; see pages 6, 7, 10, 72 and 73

PDF Portable Document Format; see pages 106 and 110

PG product goal; see pages 6 and 7

PO Product Owner; see pages 5–7, 10, 52, 72–74

PR pattern recognition; see pages 8 and 71

RAM random access memory; see pages 39 and 64

RDBMS relational database management system; see pages 43–46, 63, 66 and 154

SB sprint backlog; see pages 6, 7, 10, 72 and 73

SDM software delivery manager; see pages 10 and 79

SG sprint goal; see pages 6, 7 and 52

SM Scrum Master; see pages 5, 10 and 73

SP story point; see pages 6, 10, 11, 52, 74, 83 and 153

SQL Structured Query Language; see pages 46, 59, 61, 63, 66, 77 and 149

SSH Secure Shell; see pages 31 and 153

SUS System Usability Scale; see pages 141 and 143

UDF user-defined function; see pages 44–46

UI user interface; see pages 98, 99, 101 and 139

UML Unified Modeling Language; see page 50

URL Uniform Resource Locator; see pages 32, 38, 52, 59, 106, 112 and 129

VCS version control system; see pages 58 and 59

VM virtual machine; see pages 31, 39 and 60

Software development terminology

architecture high-level structural overview of a software system, as a design specification; see page 4

artifact a document or different byproduct that specify specific requirements, parts of the design or architecture, at greater detail; see page 5

burndown chart time-based diagram that displays lines and points that refer to certain events taking place in a sprint regarding changes to the number of story points left to work on from each point onward; see pages 6, 114 and 134

code textual files containing lines with instructions written in a programming language which perform actions that are part of a software system; see pages 4, 6, 10, 34, 37, 71, 73, 91 and 125

coverage percentage of statements or lines of code that is being executed during tests of a software product, as a measurement of how likely it is that problems and edge cases are detected; see pages 4, 6, 20, 35, 37, 73 and 91

Daily Scrum short meeting in SCRUM held every working day where the development team discusses what that have done during the sprint so far, what they are working on and possible impediments that hinder their tasks; see pages 6, 10, 73, 74 and 121

deployment installation or publication of a software product so that it is available to users; see pages 4, 10, 11 and 18

ecosystem environment in which code may be written (software development ecosystem) or a deployed product may be placed, where the developed software interacts with other systems and is dependent on a platform providing support for its functionality; see page 4

epic task that explains relationships between smaller tasks, such as user stories; see pages 6, 79 and 80

feature aspect of a software product that allows the system to perform something by providing certain functionality; see pages 4, 7 and 73, not to be confused with feature (Machine learning terminology)

guild meeting of a group of people across an entire organization with an interest in a particular topic, but available for everyone, with discussions ranging from Agile development methods to testing code and improving quality; see pages 10, 11 and 121

impediment any cause of delay and hindrance in the software development progress, which needs to be resolved before developers can continue with a certain task; see pages 5, 10 and 73

increment result of a software development cycle such as a SCRUM sprint that adheres to pre-set goals, consisting of changes from all the resolved items during that period, and may become a deployment (Potentially Shippable Product Increment) or released version, even when early in development (minimum viable product); see pages 6, 7, 11, 14, 72 and 73

maintenance regular adjustment of a software product after deployment in order to keep the product functioning in the environment in which the software is placed; see page 4

milestone moment in a software development plan that indicates an important step in the progress, usually when a new version is released or a deployment is scheduled; see pages 4 and 10

product the result of software development, fulfilling a need of users; see pages 4, 10 and 14

readiness quality of a story or other task in that it has been prepared enough during refinement meetings to be detailed enough to work on, with the team agreeing that is it not too complicated (ready for selection); see pages 6, 10, 72 and 153

refinement meeting in SCRUM to improve details of planned work for an upcoming sprint development cycle; see pages 6, 10, 71, 73 and 125

requirement specification of what a system, software and entire product should do (functional requirement) or should adhere to with regards to its environment (non-functional requirement); see pages 4, 16 and 19

retrospective meeting in SCRUM where the development team discuss internally how the previous sprint progressed and improve focus on important factors; see pages 6, 10, 14, 71–73

review meeting in SCRUM where the development team presents and discusses the results of the previous sprint with representatives of the end user, usually including a display of new functionality (demo); see pages 6, 10, 14, 71–73

sprint time span in a SCRUM development process, with specific meetings and goals, which repeats itself to work on more tasks; see pages 5, 6, 10, 72 and 73

sprint planning meeting in SCRUM to select tasks to be worked on during the next sprint development cycle; see pages 6, 10, 71 and 73

stakeholder people and parties with the most interest in a software development process, including members of the development teams, managerial roles or others in the organization, but also the end users and the client, who fulfills the role of eventual owner of a product; see pages 14, 95, 100, 102, 121 and 142

story request for a task related to developing code for a new software feature in a product and other relevant work, described in a simple format, usually in a single sentence describing a desire (user story); see pages 6, 7, 10, 72, 73 and 153

technical debt projected amount of effort, time or expenses in order to resolve a current, subpar situation so that a better solution is implemented in a software product which would require less maintenance in the future, whereas if the debt is not resolved, it will become harder to address later on, often used in the context of code style; see pages 56, 75, 79, 91 and 104

test method of comparing a software product to the specified requirements at various levels of inspection, such as small components (unit test) or interaction of systems in the software ecosystem (integration test); see pages 4, 10, 11, 20, 34, 37, 71, 73, 91, 125, 138 and 141, not to be confused with test (Machine learning terminology)

velocity metric used as a guideline for the number of story points to plan for a sprint, where the sum of the story points of all stories that were done during the past three sprints is divided by 3 (three-sprint velocity); see pages 74, 77, 78, 80, 82 and 83

Machine learning terminology

classification problem where the goal is to find a label for an unlabeled sample selected from a limited set of classes using a machine learning model (classification algorithm); see pages 7, 79–81, 84 and 91

clustering problem where the goal is to group similar samples from a data set together using a machine learning model; see pages 7, 81 and 109

data set collection of (usually different) records that describe objects, situations or events that are typically from a similar domain, with various properties mmaking up each sample record; see pages 7, 71, 79 and 90

ensemble model method to compose various machine learning algorithms together and to use their output, e.g., using a majority vote to choose the result, for solving machine learning problems; see pages 8 and 83

estimation problem where the goal is to find a label for an unlabeled sample that seems to fit the features using a machine learning or statistical model; see pages 7, 80–84 and 91

explainability quality of a machine learning algorithm, either inherent to the model used or achieved through external methods, that allows tracing back how a label or estimation was generated, for example which inputs were most relevant or which samples are most similar; see pages 8, 81, 84, 90, 152, 156 and 160

feature measurable observation about a specific sample in a data set; see pages 7, 76, 81 and 83, not to be confused with feature (Software development terminology)

feature selection process where a subset of the features from a data set are chosen based on scoring or other criteria, leading to a more refined working set; see pages 7, 72, 76, 78 and 84

label description of an object in a numerical or categorical manner, which is the goal of some machine learning problems in order to understand the data better (labeling), and when already available in the data set, is the expected outcome of the model given the sample input (target label); see page 7

model algorithm used in machine learning in order to solve a problem, such as providing a label to an object; see pages 7, 71, 76, 78, 80, 81, 83, 84 and 90

regression analysis method used to perform estimation of relationships between labels and the associated features of samples in a data set, using a function that closely fits most of the observed data points; see pages 8, 76, 103 and 109

sample entries in a data set that describe a particular object, situation or event, which may be used separately or in bulk as input for a machine learning model by selecting subsets of records (sampling); see page 7

supervised learning algorithm that is able to use labeled samples and extract statistical relations in order to learn patterns and generate numerical labels; see pages 7 and 76

test process where a portion of labeled samples from the data set (test set) is used to obtain accuracy metrics of the trained model, with a similar distribution; see pages 7, 78–81 and 84, not to be confused with test (Software development terminology)

training process where a portion of labeled samples from the data set (training set) is used to learn a model what patterns and relations between features exist in order to generate better labels in the future; see pages 7, 76, 79–81 and 84

trend outcome of a regression analysis, most typically a linear regression where the overall direction of temporal data is shown as a line, allowing for an estimation of future data points; see pages 8, 89, 103 and 109

unsupervised learning algorithm that uses unlabeled samples to extract statistical relations in order to learn patterns and similarities; see pages 7 and 109

validation process where a portion of labeled samples from the data set (validation set) is used to check if the model is well-tuned and not biased toward the samples from the training set; see pages 7, 79–81, 84 and 103

Bibliography

First referenced in Chapter 1

[1] Nayan B. Ruparelia. “Software development lifecycle models”. *ACM SIGSOFT Software Engineering Notes*, vol. 35, no. 3, 2010, pp. 8–13. DOI: 10.1145/1764810.1764814.

[2] Todd Sedano, Paul Ralph and Cécile Péraire. “Software development waste”. In: *Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering* (ICSE 2017). 2017, pp. 130–140. DOI: 10.1109/ICSE.2017.20.

[3] Agile Alliance. *Manifesto for Agile Software Development*. 2001. URL: https://agilemanifesto.org/ (visited on May 7, 2019).

[4] James A. Highsmith. *Agile Software Development Ecosystems*. Addison-Wesley, 2002.

[5] Alistair Cockburn. *Agile Software Development: The Cooperative Game*. 2nd ed. Addison-Wesley, 2007.

[6] Ken Schwaber and Jeff Sutherland. *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. 2020. URL: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf (visited on Mar. 1, 2022).

[7] Scrum.org. *The Scrum Framework Poster*. 2020. URL: https://www.scrum.org/resources/scrum-framework-poster (visited on May 29, 2021).

[8] Ken Schwaber and Jeff Sutherland. *Software in 30 Days: How Agile Managers Beat the Odds, Delight Their Customers, And Leave Competitors In the Dust*. John Wiley & Sons, 2012. DOI: 10.1002/9781119203278.

[9] Mike Cohn. *Agile Estimating and Planning*. Prentice Hall, 2005.

[10] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th ed. Pearson, 2021.

[11] Trevor Hastie, Robert Tibshirani and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer Series in Statistics. Springer, 2009. DOI: 10.1007/978-0-387-84858-7.

[12] Laurens J. P. van der Maaten, Eric O. Postma and H. Jaap van den Herik. *Dimensionality reduction: A comparative review*. Technical Report. TR 2009-005. TiCC, 2009, pp. 1–35.

[13] Kenji Kira and Larry A. Rendell. “The feature selection problem: Traditional methods and a new algorithm”. In: *Proceedings of the Tenth National Conference on Artificial Intelligence* (AAAI’92). AAAI Press, 1992, pp. 129–134.

[14] Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2016.

[15] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams. “Learning representations by back-propagating errors”. *Nature*, vol. 323, no. 6088, 1986, pp. 533–536. DOI: 10.1038/323533a0.

[16] Yoshua Bengio. “Learning deep architectures for AI”. *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, 2009, pp. 1–127. DOI: 10.1561/2200000006.

[17] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[18] Ane Blázquez-García, Angel Conde, Usue Mori and Jose A. Lozano. “A review on outlier/anomaly detection in time series data”. *ACM Computing Surveys*, vol. 54, no. 3, article 56, 2021. DOI: 10.1145/3444690.

[19] David A. Freedman. *Statistical Models: Theory and Practice*. 2nd ed. Cambridge University Press, 2009. DOI: 10.1017/CBO9780511815867.

[20] Christopher Z. Mooney. *Monte Carlo Simulation*. Quantitative Applications in the Social Sciences 116. SAGE Publications, 1997. DOI: 10.4135/9781412985116.

[21] Omer Sagi and Lior Rokach. “Ensemble learning: A survey”. *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, article 1249, 2018. DOI: 10.1002/widm.1249.

[22] Martin Shepperd, Chris Schofield and Barbara Kitchenham. “Effort estimation using analogy”. In: *Proceedings of the 18th International Conference on Software Engineering* (ICSE ’96). IEEE Computer Society, 1996, pp. 170–178. DOI: 10.1109/ICSE.1996.493413.

[23] Steven Finlay. *Predictive Analytics, Data Mining and Big Data: Myths, Misconceptions and Methods*. Business in the Digital Economy. Palgrave Macmillan, 2014. DOI: 10.1057/9781137379283.

[24] Bertrand Meyer. *Agile!: The Good, the Hype and the Ugly*. Springer, 2014. DOI: 10.1007/978-3-319-05155-0.

First referenced in Chapter 2

[25] Maarten van Steen and Andrew S. Tanenbaum. *Distributed Systems*. 3rd ed. distributed-systems.net, 2017.

[26] Gianpaolo Cugola and Alessandro Margara. “Processing flows of information: From data stream to complex event processing”. *ACM Computing Surveys*, vol. 44, no. 3, article 15, 2012. DOI: 10.1145/2187671.2187677.

[27] Philip Harrison Enslow. “What is a “distributed” data processing system?” *Computer*, vol. 11, no. 1, 1978, pp. 13–21. DOI: 10.1109/C-M.1978.217901.

[28] Franco Zambonelli, Nicholas R. Jennings and Michael J. Wooldridge. “Organisational abstractions for the analysis and design of multi-agent systems”. In: *Agent-Oriented Software Engineering* (AOSE 2000). Lecture Notes in Computer Science, vol. 1957. Springer, 2001, pp. 235–251. DOI: 10.1007/3-540-44564-1_16.

[29] Wei Ren and Yongcan Cao. *Distributed Coordination of Multi-agent Networks*. Communications and Control Engineering. Springer, 2011. DOI: 10.1007/978-0-85729-169-1.

[30] Wolfgang Reisig. *Elements of Distributed Algorithms: Modeling and Analysis with Petri Nets*. Springer, 1998. DOI: 10.1007/978-3-662-03687-7.

[31] Wil M. P. van der Aalst. “The application of Petri nets to workflow management”. *Journal of Circuits, Systems and Computers*, vol. 8, no. 1, 1998, pp. 21–66. DOI: 10.1142/S0218126698000043.

[32] MengChu Zhou and Naiqi Wu. *System Modeling and Control with Resource-Oriented Petri Nets*. CRC Press, 2010. DOI: 10.1201/9781439808856.

[33] Bruno Lopes, Mario Benevides and Edward Hermann Haeusler. “Reasoning about multi-agent systems using stochastic Petri nets”. In: *Trends in Practical Applications of Agents, Multi-Agent Systems and Sustainability*. Springer, 2015, pp. 75–86. DOI: 10.1007/978-3-319-19629-9_9.

[34] Gregor Hohpe, Bobby Woolf, Kyle Brown, Conrad F. D’Cruz, Martin Fowler, Sean Neville, Michael J. Rettig and Jonathan Simon. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. A Martin Fowler signature book. Addison-Wesley, 2004.

[35] Paolo Ceravolo et al. "Big data semantics". *Journal on Data Semantics*, vol. 7, no. 2, 2018, pp. 65–85. DOI: 10.1007/s13740-018-0086-2.

[36] Aiswarya Raj Munappy, Jan Bosch and Helena Homström Olsson. "Data pipeline management in practice: Challenges and opportunities". In: *Product-Focused Software Process Improvement* (PROFES 2020). Lecture Notes on Computer Science (Programming and Software Engineering), vol. 12562. Springer, 2020, pp. 168–184. DOI: 10.1007/978-3-030-64148-1_11.

[37] Dirk Merkel. "Docker: Lightweight Linux containers for consistent development and deployment". *Linux Journal*, vol. 2014, no. 239, article 2, 2014.

[38] Stratos Idreos, Fabian Groffen, Niels Nes, Stefan Manegold, K. Sjoerd Mullender and Martin L. Kersten. "MonetDB: Two decades of research in column-oriented database architectures". *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 35, no. 1, 2012, pp. 40–45.

[39] Gerald Carter. *LDAP System Administration: Putting Directories to Work*. O'Reilly Media, 2003.

First referenced in Chapter 3

[40] Mark Raasveldt, Pedro Holanda, Hannes Mühleisen and Stefan Manegold. "Deep integration of machine learning into column stores". In: *Proceedings of the 21st International Conference on Extending Database Technology* (EDBT). 2018, pp. 473–476. DOI: 10.5441/002/edbt.2018.50.

[41] Hannes Mühleisen, Alexander Bertram and Maarten-Jan Kallen. "Database-inspired optimizations for statistical analysis". *Journal of Statistical Software*, vol. 87, no. 4, 2018, pp. 1–20. DOI: 10.18637/jss.v087.i04.

[42] Georgios Gousios, Dominik Safaric and Joost Visser. "Streaming software analytics". In: *Proceedings of the 2016 IEEE/ACM 2nd International Workshop on Big Data Software Engineering* (BIGDSE '16). 2016, pp. 8–11. DOI: 10.1145/2896825.2896832.

[43] Mark Raasveldt. "Integrating analytics with relational databases". In: *Proceedings of the VLDB 2018 PhD Workshop co-located with the 44th International Conference on Very Large Databases* (VLDB 2018). 2018.

[44] Joseph Vinish D'Silva, Florestan De Moor and Bettina Kemme. "AIDA: Abstraction for advanced in-database analytics". *Proceedings of the VLDB Endowment*, vol. 11, no. 11, 2018, pp. 1400–1413. DOI: 10.14778/3236187.3236194.

[45] Ying Zhang, Richard Koopmanschap and Martin L. Kersten. “Love at first sight: MonetDB/TensorFlow”. In: *IEEE 34th International Conference on Data Engineering* (ICDE). 2018, pp. 1672–1672. DOI: 10.1109/ICDE.2018.00208.

[46] Jonathan Lajus and Hannes Mühleisen. “Efficient data management and statistics with zero-copy integration”. In: *Proceedings of the 26th International Conference on Scientific and Statistical Database Management* (SSDBM ’14). ACM, 2014, article 12. DOI: 10.1145/2618243.2618265.

[47] Paul Blockhaus, David Broneske, Martin Schäler, Veit Köppen and Gunter Saake. “Combining two worlds: MonetDB with multi-dimensional index structure support to efficiently query scientific data”. In: *32nd International Conference on Scientific and Statistical Database Management* (SSDBM 2020). ACM, 2020, article 29. DOI: 10.1145/3400903.3401691.

[48] Mark Raasveldt, Pedro Holanda, Tim Gubner and Hannes Mühleisen. “Fair benchmarking considered difficult: Common pitfalls in database performance testing”. In: *Proceedings of the Workshop on Testing Database Systems* (DBTest’18). ACM, 2018, article 2. DOI: 10.1145/3209950.3209955.

[49] Irene Martorelli et al. “Fungal metabarcoding data integration framework for the MycoDiversity DataBase (MDDB)”. *Journal of Integrative Bioinformatics*, vol. 17, no. 1, article 20190046, 2020. DOI: 10.1515/jib-2019-0046.

[50] Sirine Zaouali and Sonia Ayachi Ghannouchi. “Integrating quality assessment through metrics into Scrum software development”. In: *Proceedings of the 20th International Conference on New Trends in Intelligent Software Methodologies, Tools and Techniques* (SoMeT_21). Vol. 337. Frontiers in Artificial Intelligence and Applications. IOS Press. 2021, pp. 211–223. DOI: 10.3233/FAIA210021.

[51] Florencia Vega, Guillermo Rodríguez, Fabio Rocha and Rodrigo Pereira dos Santos. “Scrum Watch: A tool for monitoring the performance of Scrum-based work teams”. *Journal of Universal Computer Science*, vol. 28, no. 1, 2022, pp. 98–117. DOI: 10.3897/jucs.67593.

[52] Paulo Sérgio Santos Júnior, Monalessa Perini Barcellos, Ricardo de Almeida Falbo and João Paulo A. Almeida. “From a Scrum reference ontology to the integration of applications for data-driven software development”. *Information and Software Technology*, vol. 136, article 106570, 2021. DOI: 10.1016/j.infsof.2021.106570.

[53] Mark Raasveldt and Hannes Mühleisen. “Vectorized UDFs in column-stores”. In: *Proceedings of the 28th International Conference on Scientific and Statistical Database Management* (SSDBM ’16). ACM, 2016, article 16. DOI: 10.1145/2949689.2949703.

[54] Mark Raasveldt. “MonetDBLite: An embedded analytical database”. In: *Proceedings of the 2018 International Conference on Management of Data* (SIGMOD ’18). ACM, 2018, pp. 1837–1838. DOI: 10.1145/3183713.3183722.

[55] Peter A. Boncz, Martin L. Kersten and Stefan Manegold. “Breaking the memory wall in MonetDB”. *Communications of the ACM*, vol. 51, no. 12, 2008, pp. 77–85. DOI: 10.1145/1409360.1409380.

[56] David Gembalczyk, Felix Martin Schuhknecht and Jens Dittrich. “An experimental analysis of different key-value stores and relational databases”. In: *Datenbanksysteme für Business, Technologie und Web* (BTW2017). Gesellschaft für Informatik, 2017, pp. 351–360.

[57] Mark Raasveldt and Hannes Mühleisen. “Don’t hold my data hostage: A case for client protocol redesign”. *Proceedings of the VLDB Endowment*, vol. 10, no. 10, 2017, pp. 1022–1033. DOI: 10.14778/3115404.3115408.

[58] Fred Long, Dhruv Mohindra, Robert C. Seacord, Dean F. Sutherland and David Svoboda. *Java Coding Guidelines: 75 Recommendations for Reliable and Secure Programs*. SEI Series in Software Engineering. Pearson Education, 2013.

First referenced in Chapter 4

[59] Leon Helwerda, Frank Niessink and Fons J. Verbeek. “Conceptual process models and quantitative analysis of classification problems in Scrum software development practices”. In: *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* (IC3K 2017 - KDIR). SCITEPRESS, 2017, pp. 357–366. DOI: 10.5220/0006602803570366.

[60] Viljan Mahnič and Tomaž Hovelja. “On using planning poker for estimating user stories”. *Journal of Systems and Software*, vol. 85, no. 9, 2012, pp. 2086–2095. DOI: 10.1016/j.jss.2012.04.005.

[61] Sondra Ashmore and Kristin Runyan. *Introduction to Agile Methods*. Addison-Wesley Professional, 2014.

[62] Sergio Di Martino, Filomena Ferrucci, Carmine Gravino and Federica Sarro. “Assessing the effectiveness of approximate functional sizing approaches for effort estimation”. *Information and Software Technology*, vol. 123, article 106308, 2020. DOI: 10.1016/j.infsof.2020.106308.

[63] Zainab Masood, Rashina Hoda and Kelly Blincoe. “Real world scrum: a grounded theory of variations in practice”. *IEEE Transactions on Software Engineering*, vol. 48, no. 5, 2022, pp. 1579–1591. DOI: 10.1109/TSE.2020.3025317.

[64] Jacky Wai Keung, Barbara A. Kitchenham and David Ross Jeffery. “Analogy-X: Providing statistical inference to analogy-based software cost estimation”. *IEEE Transactions on Software Engineering*, vol. 34, no. 4, 2008, pp. 471–484. DOI: 10.1109/TSE.2008.34.

[65] Mohammad Azzeh, Daniel Neagu and Peter I. Cowling. “Analogy-based software effort estimation using fuzzy numbers”. *Journal of Systems and Software*, vol. 84, no. 2, 2011, pp. 270–284. DOI: 10.1016/j.jss.2010.09.028.

[66] Kenichi Ono, Masateru Tsunoda, Akito Monden and Kenichi Matsumoto. “Influence of outliers on analogy based software development effort estimation”. In: *Proceedings of the IEEE/ACIS 15th International Conference on Computer and Information Science* (ICIS). 2016, pp. 1–6. DOI: 10.1109/ICIS.2016.7550865.

[67] Eliane Maria De Bortoli Fávero, Roberto Pereira, Andrey Ricardo Pimentel and Dalcimar Casanova. “Analogy-based effort estimation: A systematic mapping of literature”. *INFOCOMP Journal of Computer Science*, vol. 17, no. 2, 2018, pp. 07–22.

[68] Marta Fernández-Diego, Erwin R. Méndez, Fernando González-Ladrón-de Guevara, Silvia Mara Abrahão and Emilio Insfrán. “An update on effort estimation in Agile software development: A systematic literature review”. *IEEE Access*, vol. 8, 2020, pp. 166768–166800. DOI: 10.1109/ACCESS.2020.3021664.

[69] Heejun Park and Seung Baek. “An empirical validation of a neural network model for software effort estimation”. *Expert Systems with Applications*, vol. 35, no. 3, 2008, pp. 929–937. DOI: 10.1016/j.eswa.2007.08.001.

[70] Fatima Boujida, Fatima Amazal and Ali Idri. “Neural networks based software development effort estimation: A systematic mapping study”. In: *Proceedings of the 16th International Conference on Software Technologies* (ICSOFT). SCITEPRESS, 2021, pp. 102–110. DOI: 10.5220/0010603701020110.

[71] Pantjawati Sudarmaningtyas and Rozlina Binti Mohamed. “Extended planning poker: A proposed model”. In: *Proceedings of the 7th International Conference on Information Technology, Computer, and Electrical Engineering* (ICITACEE 2020). 2020, pp. 179–184. DOI: 10.1109/ICITACEE50144.2020.9239165.

[72] Simone Porru, Alessandro Murgia, Serge Demeyer, Michele Marchesi and Roberto Tonelli. “Estimating story points from issue reports”. In: *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering* (PROMISE 2016). ACM, 2016. DOI: 10.1145/2972958.2972959.

[73] Valentina Lenarduzzi, Ilaria Lunesu, Martina Matta and Davide Taibi. “Functional size measures and effort estimation in Agile development: A replicated study”. In: *Proceedings of the 16th International Conference on Agile Processes in Software*

Engineering and Extreme Programming (XP 2015). Springer, 2015, pp. 105–116. DOI: 10.1007/978-3-319-18612-2_9.

[74] Pedro Miranda, J. Pascoal Faria, Filipe F. Correia, Ahmed Fares, Ricardo Graça and João Mendes Moreira. "An analysis of Monte Carlo simulations for forecasting software projects". In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing* (SAC '21). ACM, 2021, pp. 1550–1558. DOI: 10.1145/3412841.3442030.

[75] Howard Lei, Farnaz Ganjeizadeh, Pradeep Kumar Jayachandran and Pinar Ozcan. "A statistical analysis of the effects of Scrum and Kanban on software development projects". *Robotics and Computer-Integrated Manufacturing*, vol. 43, 2017. Special Issue: Extended Papers Selected from FAIM 2014, pp. 59–67. DOI: 10.1016/j.rcim.2015.12.001.

[76] Martin Shepperd, David Bowes and Tracy Hall. "Researcher bias: The use of machine learning in software defect prediction". *IEEE Transactions on Software Engineering*, vol. 40, no. 6, 2014, pp. 603–616. DOI: 10.1109/TSE.2014.2322358.

[77] Maria Paasivaara, Sandra Durasiewicz and Casper Lassenius. "Using Scrum in distributed Agile development: A multiple case study". In: *Proceedings of the 4th IEEE International Conference on Global Software Engineering*. 2009, pp. 195–204. DOI: 10.1109/ICGSE.2009.27.

[78] Minna Pikkarainen, Jukka Haikara, Outi Salo, Pekka Abrahamsson and Jari Still. "The impact of Agile practices on communication in software development". *Empirical Software Engineering*, vol. 13, no. 3, 2008, pp. 303–337. DOI: 10.1007/s10664-008-9065-9.

[79] Reni Kurnia, Ridi Ferdiana and Sunu Wibirama. "Software metrics classification for Agile Scrum process: A literature review". In: *Proceedings of the 2018 International Seminar on Research of Information Technology and Intelligent Systems* (ISRITI 2018). IEEE, 2018. DOI: 10.1109/isriti.2018.8864244.

[80] Wilhelm Meding. "Effective monitoring of progress of Agile software development teams in modern software companies: An industrial case study". In: *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement*. 2017, pp. 23–32. DOI: 10.1145/3143434.3143449.

[81] Prabhat Ram, Pilar Rodriguez, Markku Oivo and Silverio Martínez-Fernández. "Success factors for effective process metrics operationalization in Agile software development: A multiple case study". In: *Proceedings of the 2019 IEEE/ACM International Conference on Software and System Processes* (ICSSP 2019). IEEE, 2019, pp. 14–23. DOI: 10.1109/icssp.2019.00013.

[82] Ezequiel Scott and Dietmar Pfahl. “Using developers’ features to estimate story points”. In: *Proceedings of the 2018 International Conference on Software and System Process* (ICSSP ’18). ACM, 2018, pp. 106–110. DOI: 10.1145/3202710.3203160.

[83] Luis Almeida, Adriano Albuquerque and Plácido Pinheiro. “A multi-criteria model for planning and fine-tuning distributed Scrum projects”. In: *Proceedings of the 6th IEEE International Conference on Global Software Engineering*. 2011, pp. 75–83. DOI: 10.1109/ICGSE.2011.36.

[84] Marko Robnik-Šikonja and Igor Kononenko. “Theoretical and empirical analysis of ReliefF and RReliefF”. *Machine Learning*, vol. 53, 2003, pp. 23–69. DOI: 10.1023/A:1025667309714.

[85] Martin Tomanek and Jan Juricek. “Project risk management model based on PRINCE2 and Scrum frameworks”. *International Journal of Software Engineering & Applications*, vol. 6, no. 1, 2015, pp. 81–88. DOI: 10.5121/ijsea.2015.6107.

[86] Cyril Goutte and Eric Gaussier. “A probabilistic interpretation of precision, recall and F-score, with implication for evaluation”. In: *Advances in Information Retrieval*. Springer, 2005, pp. 345–359. DOI: 10.1007/978-3-540-31865-1_25.

[87] Shashank Mouli Satapathy and Santanu Kumar Rath. “Empirical assessment of machine learning models for Agile software development effort estimation using story points”. *Innovations in Systems and Software Engineering*, vol. 13, no. 2, 2017, pp. 191–200. DOI: 10.1007/s11334-017-0288-z.

[88] Subhra Sankar Dhar, Biman Chakraborty and Probal Chaudhuri. “Comparison of multivariate distributions using quantile-quantile plots and related tests”. *Bernoulli*, vol. 20, no. 3, 2014, pp. 1484–1506. DOI: 10.3150/13-BEJ530.

[89] Sebastian Baltes and Paul Ralph. “Sampling in software engineering research: a critical review and guidelines”. *Empirical Software Engineering*, vol. 27, article 94, 2022. DOI: 10.1007/s10664-021-10072-8.

First referenced in Chapter 5

[90] Edward R. Tufte. *Envisioning Information*. 2nd ed. Graphics Press, 1998.

[91] Min Chen, David Ebert, Hans Hagen, Robert S. Laramee, Robert van Liere, Kwan-Liu Ma, William Ribarsky, Gerik Scheuermann and Deborah Silver. “Data, information, and knowledge in visualization”. *IEEE Computer Graphics and Applications*, vol. 29, no. 1, 2008, pp. 12–19. DOI: 10.1109/MCG.2009.6.

[92] Usama Fayyad, Georges G. Grinstein and Andreas Wierse. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers Inc., 2001.

[93] David P. Tegarden. "Business information visualization". *Communications of the Association for Information Systems*, vol. 1, article 4, 1999. DOI: `10.17705/1cais.00104`.

[94] Ben Shneiderman, Catherine Plaisant, Maxine S. Cohen, Steven Jacobs, Niklas Elmqvist and Nicholas Diakopoulos. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 6th ed. Pearson, 2016, pp. 66–82.

[95] Iris Vessey and Dennis Galletta. "Cognitive fit: An empirical study of information acquisition". *Information Systems Research*, vol. 2, no. 1, 1991, pp. 63–84. DOI: `10.1287/isre.2.1.63`.

[96] Joseph K. Nuamah, Younho Seong, Steven Jiang, Eui Park and Daniel Mountjoy. "Evaluating effectiveness of information visualizations using cognitive fit theory: A neuroergonomics approach". *Applied Ergonomics*, vol. 88, article 103173, 2020. DOI: `10.1016/j.apergo.2020.103173`.

[97] Anshul Vikram Pandey, Anjali Manivannan, Oded Nov, Margaret Satterthwaite and Enrico Bertini. "The persuasive power of data visualization". *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, 2014, pp. 2211–2220. DOI: `10.1109/TVCG.2014.2346419`.

[98] Julia Paredes, Craig Anslow and Frank Maurer. "Information visualization for Agile software development teams". In: *Proceedings of the Second IEEE Working Conference on Software Visualization* (VISSOFT 2014). IEEE, 2014, pp. 157–166. DOI: `10.1109/vissoft.2014.32`.

[99] Antonio González-Torres, Francisco J. García-Peñalvo, Roberto Therón-Sánchez and Ricardo Colomo-Palacios. "Knowledge discovery in software teams by means of evolutionary visual software analytics". *Science of Computer Programming*, vol. 121, 2016. Special Issue on Knowledge-based Software Engineering, pp. 55–74. DOI: `10.1016/j.scico.2015.09.005`.

[100] Nesib Tekin, Mehmet Kosa, Murat Yilmaz, Paul Clarke and Vahid Garousi. "Visualization, monitoring and control techniques for use in Scrum software development: An Analytic Hierarchy Process approach". In: *Systems, Software and Services Process Improvement*. Springer, 2020, pp. 45–57. DOI: `10.1007/978-3-030-56441-4_4`.

[101] Martin J. Eppler and Sabrina Bresciani. "Visualization in management: From communication to collaboration. A response to Zhang". *Journal of Visual Languages & Computing*, vol. 24, no. 2, 2013, pp. 146–149. DOI: `10.1016/j.jvlc.2012.11.003`.

[102] Evanthia Dimara and Charles Perin. "What is interaction for data visualization?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, 2020, pp. 119–129. DOI: `10.1109/TVCG.2019.2934283`.

[103] Riccardo Mazza. *Introduction to Information Visualization*. Springer, 2009. DOI: 10.1007/978-1-84800-219-7.

[104] Juuso Koponen and Jonatan Hildén. *Data Visualization Handbook*. Art + Design + Architecture. Aalto University, 2019.

[105] Bang Wong. "Points of view: Color blindness". *Nature Methods*, vol. 8, no. 6, 2011, pp. 441–441. DOI: 10.1038/nmeth.1618.

[106] Georges Grinstein, Alfred Kobsa, Catherine Plaisant and John T. Stasko. "Which comes first, usability or utility?" In: *Proceedings of the IEEE Conference on Visualization* (VIS 2003). IEEE, 2003, pp. 605–606. DOI: 10.1109/visual.2003.1250426.

[107] Jeffrey Heer, Stuart K. Card and James A. Landay. "prefuse: A toolkit for interactive information visualization". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '05). ACM, 2005, pp. 421–430. DOI: 10.1145/1054972.1055031.

[108] Stuart K. Card, Jock D. Mackinlay and Ben Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Interactive Technologies. Morgan Kaufmann, 1999.

[109] Ben Shneiderman. "The eyes have it: A task by data type taxonomy for information visualizations". In: *The Craft of Information Visualization*. Morgan Kaufmann, 2003, pp. 364–371. DOI: https://doi.org/10.1016/B978-155860915-0/50046-9.

[110] Jakob Nielsen. *Usability Engineering*. Morgan Kaufman, 1993. DOI: 10.1016/C2009-0-21512-1.

[111] Michael Bostock, Vadim Ogievetsky and Jeffrey Heer. "$\mathbb{D}^3$: Data-Driven Documents". *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, 2011, pp. 2301–2309. DOI: 10.1109/TVCG.2011.185.

[112] Patrick Riehmann, Manfred Hanfler and Bernd Froehlich. "Interactive Sankey diagrams". In: *IEEE Symposium on Information Visualization* (INFOVIS 2005). 2005, pp. 233–240. DOI: 10.1109/INFVIS.2005.1532152.

[113] Amy N. Langville and Carl D. Meyer. *Who's #1?: The Science of Rating and Ranking*. Princeton University Press, 2012. DOI: 10.1515/9781400841677.

[114] Josh Barnes and Piet Hut. "A hierarchical $O(N \log N)$ force-calculation algorithm". *Nature*, vol. 324, no. 6096, 1986, pp. 446–449. DOI: 10.1038/324446a0.

[115] Loup Verlet. "Computer "experiments" on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules". *Physical Review*, vol. 159, no. 1, article 98, 1967, pp. 98–103. DOI: 10.1103/PhysRev.159.98.

[116] Emden R. Gansner and Stephen C. North. “An open graph visualization system and its applications to software engineering”. *Software: Practice and Experience*, vol. 30, no. 11, 2000. Special Issue: Discrete algorithm engineering, pp. 1203–1233. DOI: 10.1002/1097-024X(200009)30:11<1203::AID-SPE338>3.0.CO;2-N.

[117] Cas H. J. Dekkers. “Designing information visualizations for generating business value in Agile software development”. Master’s thesis. LIACS, Leiden University, 2021.

[118] Laurens C. Groeneveld. “Visalization of patterns in Scrum software development”. Bachelor’s thesis. LIACS, Leiden University, 2017.

[119] Donald A. Norman. *The Design of Everyday Things*. Revised and Expanded Edition. Originally published as *The Psychology of Everyday Things*. Perseus Books, 2013.

[120] Catherine Plaisant. “The challenge of information visualization evaluation”. In: *Proceedings of the Working Conference on Advanced Visual Interfaces* (AVI ’04). ACM, 2004, pp. 109–116. DOI: 10.1145/989863.989880.

[121] Michael Behrisch et al. “Quality metrics for information visualization”. *Computer Graphics Forum*, vol. 37, no. 3, 2018, pp. 625–662. DOI: 10.1111/cgf.13446.

[122] Evanthia Dimara, Anastasia Bezerianos and Pierre Dragicevic. “Conceptual and methodological issues in evaluating multidimensional visualizations for decision support”. *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, 2018, pp. 749–759. DOI: 10.1109/TVCG.2017.2745138.

Technical resources

[I] Atlassian. *Jira: Issue & project tracking software*. URL: https://www.atlassian.com/software/jira.

[II] Scott Chacon et al. *Git*. URL: https://git-scm.com/.

[III] GitLab. *The most-comprehensive AI-powered DevSecOps platform*. URL: https://about.gitlab.com/.

[IV] Continuous Delivery Foundation. *Jenkins*. URL: https://www.jenkins.io/.

[V] SonarSource. *Code quality, security & static analysis tool with SonarQube*. URL: https://www.sonarsource.com/products/sonarqube/.

[VI] ICTU. *Quality-time: Software quality monitoring for teams and projects*. URL: https://github.com/ICTU/quality-time.

[VII] ICTU. *BigBoat: An open-source container and CI/CD ecosystem*. URL: https://github.com/bigboat-io.

[VIII] Microsoft. *Azure DevOps Server*. URL: https://azure.microsoft.com/en-us/products/devops/server/.

[IX] The GnuPG Project. *The GNU Privacy Guard*. URL: https://www.gnupg.org/.

[X] The R Foundation. *The R project for statistical computing*. URL: https://www.r-project.org/.

[XI] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[XII] Docker. *Docker Compose overview*. URL: https://docs.docker.com/compose/.

[XIII] GitHub. *GitHub: Let's build from here*. URL: https://github.com/.

[XIV] Apache Software Foundation. *Apache Subversion*. URL: https://subversion.apache.org/.

[XV] TOPdesk. *IT Service Management Platform*. URL: https://www.topdesk.com/en/.

[XVI] Oracle. *MySQL Workbench*. URL: https://dev.mysql.com/doc/workbench/en/.

[XVII] Lance Andersen. *JDBC 4.2 Specification*. Oracle, 2014. URL: https://download.oracle.com/otn-pub/jcp/jdbc-4_2-mrel2-spec/jdbc4.2-fr-spec.pdf.

[XVIII] Transaction Processing Performance Council. *TPC Benchmark H (Decision Support) Standard Specification*. Apr. 28, 2022. URL: https://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v3.0.1.pdf.

[XIX] Leon Helwerda. *Grip on Software sprint features*. 2024. DOI: 10.5281/zenodo.10878529. URL: https://gros.liacs.nl/combined/prediction/api/v1/dataset.

[XX] Eibe Frank et al. *ARFF Format*. URL: https://waikato.github.io/weka-wiki/formats_and_processing/arff/.

[XXI] Jeremy Thomas. *Bulma: Free, open source, and modern CSS framework based on Flexbox*. URL: https://bulma.io/.

[XXII] Orit Golowinski. *Understand how your teams adopt DevOps with DevOps reports*. Dec. 15, 2021. URL: https://about.gitlab.com/blog/2021/12/15/devops-adoption/.

[XXIII] Mike Bostock. *d3-force: Force-directed graph layout using velocity Verlet integration*. URL: `https://github.com/d3/d3-force`.

[XXIV] The Graphviz Authors. *DOT Language*. URL: `https://www.graphviz.org/doc/info/lang.html`.

[XXV] Magnus Jacobsson. *d3-graphviz: Graphviz DOT rendering and animated transitions using D3*. URL: `https://github.com/magjac/d3-graphviz`.

[XXVI] WHATWG. *HTML Standard: Web workers*. Mar. 17, 2022. URL: `https://html.spec.whatwg.org/multipage/workers.html`.

[XXVII] Justin Palmer. *Introducing Contributions*. Jan. 7, 2013. URL: `https://github.blog/2013-01-07-introducing-contributions/`.

[XXVIII] Koninklijk Nederlands Meteorologisch Instituut (KNMI). *Meteo data - daily quality controlled climate data*. URL: `https://dataplatform.knmi.nl/dataset/etmaalgegevensknmistations-1`.

[XXIX] Meta. *React*. URL: `https://react.dev/`.

[XXX] Dan Abramov. *Redux - A predictable state container for JavaScript apps*. URL: `https://redux.js.org/`.

[XXXI] Mark Otto, Jacob Thorton and Bootstrap contributors. *Bootstrap*. URL: `https://getbootstrap.com/`.

[XXXII] Deque Systems. *axe: Accessiblity Testing Tools and Software*. URL: `https://www.deque.com/axe/`.

[XXXIII] Andrew Kirkpatrick, Joshue O'Connor, Alastair Campbell and Michael Cooper. *Web Content Accessibility Guidelines (WCAG) 2.1*. W3C Recommendation. June 5, 2018. URL: `https://www.w3.org/TR/WCAG21/`.

[XXXIV] Joanmarie Diggs, James Nurthen and Michael Cooper. *Accessible Rich Internet Applications (WAI-ARIA) 1.2*. W3C Candidate Recommendation Draft. Dec. 8, 2021. URL: `https://www.w3.org/TR/wai-aria-1.2/`.

Appendices

Appendix A

Code repositories of the Grip on Software pipeline

Accompanying Chapter 2

The references listed here are supplemental to the technical resources found in the bibliography. We provide these separate from the bibliography, given their nature of being contributions in addition to—and in support of—our research. The references indicate locations of code repositories that contain implementations, documentation and tests for the components of the GROS pipeline used throughout our research. In Section 2.3.2, we provide descriptions and further details for each of the code repositories.

[a] Leon Helwerda. *Modules used to gather data from different data sources in software development processes*. ICTU and Leiden University. DOI: 10.5281/zenodo.10911862. URL: https://github.com/grip-on-software/data-gathering.

[b] Leon Helwerda. *Web-based data gathering agent configuration*. ICTU and Leiden University. DOI: 10.5281/zenodo.11115708. URL: https://github.com/grip-on-software/agent-config.

[c] Leon Helwerda. *Data gathering agent status web application*. ICTU and Leiden University. DOI: 10.5281/zenodo.12533335. URL: https://github.com/grip-on-software/status-dashboard.

[d] Leon Helwerda, Enrique Larios Vargas, Thomas Helling and Thomas Prikkel. *Importer of gathered data into a MonetDB database*. ICTU and Leiden University. DOI: 10.5281/zenodo.12583196. URL: https://github.com/grip-on-software/monetdb-import.

[e] Leon Helwerda. *Export tables from a MonetDB database for backups or exchanges*. ICTU and Leiden University. DOI: 10.5281/zenodo.12723675. URL: https://github.com/grip-on-software/monetdb-dumper.

[f] Leon Helwerda. *Tools for securely uploading files to a remote server via HTTPS and GPG*. ICTU and Leiden University. DOI: 10.5281/zenodo.12773659. URL: https://github.com/grip-on-software/export-exchange.

[g] Leon Helwerda. *Encrypted file upload server*. ICTU and Leiden University. DOI: 10.5281/zenodo.12784820. URL: https://github.com/grip-on-software/upload.

[h] Leon Helwerda. *Requesting (anonymized/aggregate) data from a filled MonetDB database and processing for data mining or visualization*. ICTU and Leiden University. DOI: 10.5281/zenodo.12935240. URL: https://github.com/grip-on-software/data-analysis.

[i] Leon Helwerda. *Algorithms to predict, classify and analyze features and labels of Scrum data*. ICTU and Leiden University. DOI: 10.5281/zenodo.12942716. URL: https://github.com/grip-on-software/prediction.

[j] Leon Helwerda. *Integrated visualization hub*. ICTU and Leiden University. DOI: 10.5281/zenodo.13208936. URL: https://github.com/grip-on-software/visualization-site.

[k] Leon Helwerda. *Common visualization UI fragments*. ICTU and Leiden University. URL: https://github.com/grip-on-software/visualization-ui.

[l] Leon Helwerda. *Dynamic sprint report generator in comparison visualization formats*. ICTU and Leiden University. DOI: 10.5281/zenodo.13208969. URL: https://github.com/grip-on-software/sprint-report.

[m] Leon Helwerda. *Human-readable output of sprint predictions*. ICTU and Leiden University. DOI: 10.5281/zenodo.13209623. URL: https://github.com/grip-on-software/prediction-site.

[n] Leon Helwerda. *Interactive visualization of temporal data from a software development process*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220620. URL: https://github.com/grip-on-software/timeline.

[o] Leon Helwerda and Laurens C. Groeneveld. *Project statistics as a leaderboard*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220623. URL: https://github.com/grip-on-software/leaderboard.

[p] Leon Helwerda and Laurens C. Groeneveld. *Graph of relations between project members and the projects they work on*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220626. URL: https://github.com/grip-on-software/collaboration-graph.

[q] Leon Helwerda. *Flowchart display of story status*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220648. URL: https://github.com/grip-on-software/process-flow.

[r] Leon Helwerda and Laurens C. Groeneveld. *Visualization of project commit activity over time*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220681. URL: https://github.com/grip-on-software/heatmap.

[s] Leon Helwerda and Laurens C. Groeneveld. *BigBoat platform reliability graphs*. ICTU and Leiden University. DOI: 10.5281/zenodo.13220696. URL: https://github.com/grip-on-software/bigboat-status.

[t] Cas H. J. Dekkers. *Effort burndown chart for product backlogs*. ICTU and Leiden University. URL: https://github.com/grip-on-software/backlog-burndown.

[u] Cas H. J. Dekkers. *Progression inspection chart for product backlogs*. ICTU and Leiden University. URL: https://github.com/grip-on-software/backlog-progression.

[v] Cas H. J. Dekkers. *Issue relationship chart for product backlogs*. ICTU and Leiden University. URL: `https://github.com/grip-on-software/backlog-relationship`.

[w] Leon Helwerda. *Deployment web application*. ICTU and Leiden University. DOI: `10.5281/zenodo.12571035`. URL: `https://github.com/grip-on-software/deployer`.

[x] Leon Helwerda. *Web application framework for building authenticated services with templating support*. ICTU and Leiden University. DOI: `10.5281/zenodo.11580150`. URL: `https://github.com/grip-on-software/server-framework`.

[y] Leon Helwerda. *Cleanup of Docker, Jenkins and SonarQube services based on build state*. ICTU and Leiden University. URL: `https://github.com/grip-on-software/jenkins-cleanup`.

[z] Leon Helwerda. *Collect JavaScript coverage information during a test run*. ICTU and Leiden University. URL: `https://github.com/grip-on-software/coverage-collector`.

Appendix B

Queries used in database performance experiments

Accompanying Chapter 3

```
SELECT ${f(join_cols, "sprint_metrics")}, COUNT(*) AS
    num_metrics
FROM (
    SELECT DISTINCT ${f(join_cols, "metric_value")},
        metric_value.metric_id
    FROM gros.metric_value
    JOIN gros.${t("sprint")} ON ${j(join_cols,
        "metric_value", "sprint")}
    WHERE metric_value.value <> -1
) AS sprint_metrics
${g(join_cols, "sprint_metrics")}
```

(a) Template

```
SELECT sprint_metrics.project_id,
    sprint_metrics.sprint_id, COUNT(*) AS num_metrics
FROM (
    SELECT DISTINCT metric_value.project_id,
        metric_value.sprint_id, metric_value.metric_id
    FROM gros.metric_value
    JOIN gros.sprint ON metric_value.project_id =
        sprint.project_id AND metric_value.sprint_id =
        sprint.sprint_id
    WHERE metric_value.value <> -1
) AS sprint_metrics
GROUP BY sprint_metrics.project_id,
    sprint_metrics.sprint_id
```

(b) Compiled

Figure B.1: All metrics (original query)

```
SELECT ${f(join_cols, "sprint_metrics")}, COUNT(*) AS
    num_metrics
FROM (
    SELECT DISTINCT ${f(join_cols, "metric_value")},
        metric_value.metric_id
    FROM gros.metric_value
    WHERE metric_value.value <> -1 AND
        metric_value.sprint_id <> 0
) AS sprint_metrics
${g(join_cols, "sprint_metrics")}
```

(a) Template

```
SELECT sprint_metrics.project_id,
    sprint_metrics.sprint_id, COUNT(*) AS num_metrics
FROM (
    SELECT DISTINCT metric_value.project_id,
        metric_value.sprint_id, metric_value.metric_id
    FROM gros.metric_value
    WHERE metric_value.value <> -1 AND
        metric_value.sprint_id <> 0
) AS sprint_metrics
GROUP BY sprint_metrics.project_id,
    sprint_metrics.sprint_id
```

(b) Compiled

Figure B.2: All metrics (refined query)

```
SELECT ${f(join_cols, "sprint_metrics")}, COUNT(*) AS
    num_red_metrics
FROM (
    SELECT DISTINCT ${f(join_cols, "metric_value")},
        metric_value.metric_id
    FROM gros.metric_value
    JOIN gros.${t("sprint")} ON ${j(join_cols,
        "metric_value", "sprint")}
    WHERE metric_value.category = 'red'
) AS sprint_metrics
${g(join_cols, "sprint_metrics")}
```

(a) Template

```
SELECT sprint_metrics.project_id, sprint_metrics.sprint_id,
    COUNT(*) AS num_red_metrics
FROM (
    SELECT DISTINCT metric_value.project_id,
        metric_value.sprint_id, metric_value.metric_id
    FROM gros.metric_value
    JOIN gros.sprint ON metric_value.project_id =
        sprint.project_id AND metric_value.sprint_id =
        sprint.sprint_id
    WHERE metric_value.category = 'red'
) AS sprint_metrics
GROUP BY sprint_metrics.project_id, sprint_metrics.sprint_id
```

(b) Compiled

Figure B.3: Red metrics (original query)

```
SELECT ${f(join_cols, "sprint_metrics")}, COUNT(*) AS
    num_red_metrics
FROM (
    SELECT DISTINCT ${f(join_cols, "metric_value")},
        metric_value.metric_id
    FROM gros.metric_value
    WHERE metric_value.category = 'red' AND
        metric_value.sprint_id <> 0
) AS sprint_metrics
${g(join_cols, "sprint_metrics")}
```

(a) Template

```
SELECT sprint_metrics.project_id, sprint_metrics.sprint_id,
    COUNT(*) AS num_red_metrics
FROM (
    SELECT DISTINCT metric_value.project_id,
        metric_value.sprint_id, metric_value.metric_id
    FROM gros.metric_value
    WHERE metric_value.category = 'red' AND
        metric_value.sprint_id <> 0
) AS sprint_metrics
GROUP BY sprint_metrics.project_id, sprint_metrics.sprint_id;
```

(b) Compiled

Figure B.4: Red metrics (refined query)

```
SELECT ${f(join_cols, "team_spirit")},
    AVG(metric_value.value) AS team_spirit
FROM gros.metric_value, (
    SELECT ${f(join_cols, "metric_value")},
        metric_value.metric_id, MAX(metric_value.date) AS
        max_date
    FROM gros.metric_value
    JOIN gros.metric
    ON metric_value.metric_id = metric.metric_id
    JOIN gros.${t("sprint")}
    ON ${j(join_cols, "metric_value", "sprint")}
    WHERE metric_value.value <> -1
    AND metric.base_name = 'TeamSpirit'
    ${g(join_cols, "metric_value")}, metric_value.metric_id
) AS team_spirit
WHERE metric_value.date = team_spirit.max_date AND
    metric_value.metric_id = team_spirit.metric_id
${g(join_cols, "team_spirit")}
```

(a) Template

```
SELECT team_spirit.project_id, team_spirit.sprint_id,
    AVG(metric_value.value) AS team_spirit
FROM gros.metric_value, (
    SELECT metric_value.project_id, metric_value.sprint_id,
        metric_value.metric_id, MAX(metric_value.date) AS
        max_date
    FROM gros.metric_value
    JOIN gros.metric
    ON metric_value.metric_id = metric.metric_id
    JOIN gros.sprint
    ON metric_value.project_id = sprint.project_id AND
        metric_value.sprint_id = sprint.sprint_id
    WHERE metric_value.value <> -1
    AND metric.base_name = 'TeamSpirit'
    GROUP BY metric_value.project_id,
        metric_value.sprint_id, metric_value.metric_id
) AS team_spirit
WHERE metric_value.date = team_spirit.max_date and
    metric_value.metric_id = team_spirit.metric_id
GROUP BY team_spirit.project_id, team_spirit.sprint_id
```

(b) Compiled

Figure B.5: Team spirit (original query)

```
SELECT ${f(join_cols, "team_spirit")}, MAX(value) AS team_spirit
FROM (
    SELECT ${f(join_cols, "metric_value")}, metric_value.value,
        MAX(metric_value.date) AS end_date, ROW_NUMBER() OVER (
        PARTITION BY ${f(join_cols, "metric_value")}
        ORDER BY ${f(join_cols, "metric_value")},
            MAX(metric_value.date) DESC
    ) AS rev_row FROM gros.metric_value
    JOIN gros.metric
    ON metric_value.metric_id = metric.metric_id
    WHERE metric.base_name = 'TeamSpirit' AND metric.domain_name
        <> '' AND metric_value.sprint_id <> 0
    AND metric_value.value > -1
    ${g(join_cols, "metric_value")}, metric_value.value
) AS team_spirit
WHERE rev_row = 1
${g(join_cols, "team_spirit")}
```

(a) Template

```
SELECT team_spirit.project_id, team_spirit.sprint_id, MAX(value)
    AS team_spirit
FROM (
    SELECT metric_value.project_id, metric_value.sprint_id,
        metric_value.value, MAX(metric_value.date) AS end_date,
        ROW_NUMBER() OVER (
        PARTITION BY metric_value.project_id,
            metric_value.sprint_id
        ORDER BY metric_value.project_id, metric_value.sprint_id,
            MAX(metric_value.date) DESC
    ) AS rev_row FROM gros.metric_value
    JOIN gros.metric
    ON metric_value.metric_id = metric.metric_id
    WHERE metric.base_name = 'TeamSpirit' AND metric.domain_name
        <> '' AND metric_value.sprint_id <> 0
    AND metric_value.value > -1
    GROUP BY metric_value.project_id, metric_value.sprint_id,
        metric_value.value
) AS metric_team_spirit
WHERE rev_row = 1
GROUP BY metric_team_spirit.project_id,
    metric_team_spirit.sprint_id
```

(b) Compiled

Figure B.6: Team spirit (refined query)

```
SELECT ${f(join_cols, "issue", mask=1)}, ${s(issue_key)} AS key,
    MAX(${f(join_cols, "sprint", mask=2, alias=T,
    sprint="interval_sprint")}) AS ${f(join_cols, "", mask=2, alias=F)},
    MAX(${s(story_points)}) AS story_points, MAX(${s(fix_version)}) AS
    fixversion
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS older_issue
ON ${j(issue_next_changelog, "issue", "older_issue")}
LEFT JOIN gros.${t("sprint")}
ON ${j(join_cols, "issue", "sprint")}
JOIN gros.${t("sprint")} AS interval_sprint
ON ${j(join_cols, "issue", "interval_sprint", 1)}
WHERE (${f(join_cols, "sprint", mask=2, alias="alias")} IS NULL
    OR ${s(sprint_open)} >= ${s(sprint_open, sprint="interval_sprint")}
)
AND ${s(issue_not_done)}
AND ${s(issue_backlog)}
AND ${t("issue")}.updated > ${s(sprint_open, sprint="interval_sprint")}
AND (${t("older_issue")}.changelog_id IS NULL ${s(filter_inverse,
    issue="older_issue", cond_op="OR")})
${g(join_cols, "issue", f("issue_key"), mask=1)}
```

(a) Template

```
SELECT issue.project_id, issue.key AS key, MAX(interval_sprint.sprint_id)
    AS sprint_id, MAX(CASE WHEN issue.story_points IN (-5, -1, 99, 100,
    122, 999) THEN 0 ELSE issue.story_points END) AS story_points,
    MAX(issue.fixversion) AS fixversion
FROM gros.issue
LEFT JOIN gros.issue AS older_issue
ON issue.issue_id = older_issue.issue_id AND issue.changelog_id =
    older_issue.changelog_id + 1
LEFT JOIN gros.sprint
ON issue.project_id = sprint.project_id AND issue.sprint_id =
    sprint.sprint_id
JOIN gros.sprint AS interval_sprint
ON issue.project_id = interval_sprint.project_id
WHERE (sprint.sprint_id IS NULL
    OR COALESCE(CAST(sprint.start_date AS TIMESTAMP), CURRENT_TIMESTAMP())
        >= COALESCE(CAST(interval_sprint.start_date AS TIMESTAMP),
        CURRENT_TIMESTAMP())
)
AND COALESCE(issue.resolution, 0) NOT IN (1, 10000) AND
    COALESCE(issue.status, 0) NOT IN (6, 10008)
AND issue."type" = 7
AND issue.updated > COALESCE(CAST(interval_sprint.start_date AS TIMESTAMP),
    CURRENT_TIMESTAMP())
AND (older_issue.changelog_id IS NULL)
GROUP BY issue.project_id, issue.issue_id, issue.key
```

(b) Compiled

Figure B.7: Backlog added points (original query)

```
SELECT ${f(join_cols, "issue", mask=1)}, ${s(issue_key)} AS
    key, MAX(${f(join_cols, "sprint", mask=2, alias=T,
    sprint="interval_sprint")}) AS ${f(join_cols, "",
    mask=2, alias=F)}, MAX(${s(story_points)}) AS
    story_points, MAX(${s(fix_version)}) AS fixversion
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS older_issue
ON ${j(issue_next_changelog, "issue", "older_issue")}
JOIN gros.${t("sprint")} AS interval_sprint
ON ${j(join_cols, "issue", "interval_sprint", 1)}
AND interval_sprint.sprint_id IN (${filter_sprint_ids})
AND ${t("issue")}.updated > ${s(sprint_open,
    sprint="interval_sprint")}
WHERE ${s(issue_not_done)}
AND ${s(issue_backlog)}
AND (${t("older_issue")}.changelog_id IS NULL
    ${s(filter_inverse, issue="older_issue", cond_op="OR")})
${g(join_cols, "issue", f("issue_key"), mask=1)}
```

(a) Template

```
SELECT issue.project_id, issue.key AS key,
    MAX(interval_sprint.sprint_id) AS sprint_id,
    MAX(CASE WHEN issue.story_points IN (-5, -1, 99, 100,
        122, 999) THEN 0 ELSE issue.story_points END) AS
        story_points,
    MAX(issue.fixversion) AS fixversion
FROM gros.issue
LEFT JOIN gros.issue AS older_issue
ON issue.issue_id = older_issue.issue_id AND
    issue.changelog_id = older_issue.changelog_id + 1
JOIN gros.sprint AS interval_sprint
ON issue.project_id = interval_sprint.project_id
AND interval_sprint.sprint_id IN (...)
AND issue.updated > COALESCE(CAST(interval_sprint.start_date
    AS TIMESTAMP), CURRENT_TIMESTAMP())
WHERE COALESCE(issue.resolution, 0) NOT IN (1, 10000) AND
    COALESCE(issue.status, 0) NOT IN (6, 10008)
AND issue."type" = 7
AND (older_issue.changelog_id IS NULL)
GROUP BY issue.project_id, issue.issue_id, issue.key
```

(b) Compiled

Figure B.8: Backlog added points (refined query)

```
SELECT ${f(join_cols, "sprint", alias=T, sprint="in_sprint")},
    ${t("issue")}.epic AS key, COUNT(*) AS epic_children,
    SUM(${s(story_points)}) AS story_points
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS newer_issue
ON ${j(issue_next_changelog, "newer_issue", "issue")}
LEFT JOIN gros.${t("sprint")} ON ${j(join_cols, "issue", "sprint")}
JOIN gros.${t("sprint")} AS in_sprint
ON ${j(join_cols, "issue", "in_sprint", 1)}
WHERE ${t("issue")}.epic IS NOT NULL
AND (${f(join_cols, "sprint", mask=2, alias="alias")} IS NULL OR
    ${s(sprint_open)} >= ${s(sprint_close, sprint="in_sprint")})
AND ${s(issue_story)} AND ${s(issue_not_done)}
AND ${t("issue")}.updated <= ${s(sprint_close, sprint="in_sprint")}
AND (newer_issue.updated IS NULL OR newer_issue.updated > ${s(sprint_close,
    sprint="in_sprint")})
${g(join_cols, "sprint", sprint="in_sprint")}, ${t("issue")}.epic
```

(a) Template

```
SELECT in_sprint.project_id, in_sprint.sprint_id, issue.epic AS key,
    COUNT(*) AS epic_children, SUM(CASE WHEN issue.story_points IN (-5, -1,
    99, 100, 122, 999) THEN 0 ELSE issue.story_points END) AS story_points
FROM gros.issue
LEFT JOIN gros.issue AS newer_issue
ON newer_issue.issue_id = issue.issue_id AND newer_issue.changelog_id =
    issue.changelog_id + 1
LEFT JOIN gros.sprint ON issue.project_id = sprint.project_id AND
    issue.sprint_id = sprint.sprint_id
JOIN gros.sprint AS in_sprint
ON issue.project_id = in_sprint.project_id
WHERE issue.epic IS NOT NULL
AND (sprint.sprint_id IS NULL OR COALESCE(CAST(sprint.start_date AS
    TIMESTAMP), CURRENT_TIMESTAMP()) >= CASE WHEN in_sprint.complete_date
    IS NOT NULL AND CAST(in_sprint.complete_date AS DATE) <
    CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date ELSE
    in_sprint.end_date END)
AND issue."type" = 7 AND COALESCE(issue.resolution, 0) NOT IN (1, 10000)
    AND COALESCE(issue.status, 0) NOT IN (6, 10008)
AND issue.updated <= CASE WHEN in_sprint.complete_date IS NOT NULL AND
    CAST(in_sprint.complete_date AS DATE) < CAST(in_sprint.end_date AS
    DATE) THEN in_sprint.complete_date ELSE in_sprint.end_date END
AND (newer_issue.updated IS NULL OR newer_issue.updated > CASE WHEN
    in_sprint.complete_date IS NOT NULL AND CAST(in_sprint.complete_date AS
    DATE) < CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date
    ELSE in_sprint.end_date END)
GROUP BY in_sprint.project_id, in_sprint.sprint_id, issue.epic
```

(b) Compiled

Figure B.9: Backlog epic points (original query)

```
SELECT ${f(join_cols, "sprint", alias=T, sprint="in_sprint")},
    ${t("issue")}.epic AS key, COUNT(*) AS epic_children,
    SUM(${s(story_points)}) AS story_points
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS newer_issue
ON ${j(issue_next_changelog, "newer_issue", "issue")}
LEFT JOIN gros.${t("sprint")} ON ${j(join_cols, "issue", "sprint")}
JOIN gros.${t("sprint")} AS in_sprint
ON ${j(join_cols, "issue", "in_sprint", 1)}
AND in_sprint.sprint_id IN (${filter_sprint_ids})
AND ${t("issue")}.updated <= ${s(sprint_close, sprint="in_sprint")}
AND COALESCE(newer_issue.updated, ${s(sprint_close, sprint="in_sprint")})
    >= ${s(sprint_close, sprint="in_sprint")}
WHERE ${t("issue")}.epic IS NOT NULL AND (${f(join_cols, "sprint", mask=2,
    alias="alias")} IS NULL OR ${s(sprint_open)} >= ${s(sprint_close,
    sprint="in_sprint")}) AND ${s(issue_story)} AND ${s(issue_not_done)}
${g(join_cols, "sprint", sprint="in_sprint")}, ${t("issue")}.epic
```

(a) Template

```
SELECT in_sprint.project_id, in_sprint.sprint_id, issue.epic AS key,
    COUNT(*) AS epic_children, SUM(CASE WHEN issue.story_points IN (-5, -1,
    99, 100, 122, 999) THEN 0 ELSE issue.story_points END) AS story_points
FROM gros.issue
LEFT JOIN gros.issue AS newer_issue
ON newer_issue.issue_id = issue.issue_id AND newer_issue.changelog_id =
    issue.changelog_id + 1
LEFT JOIN gros.sprint ON issue.project_id = sprint.project_id AND
    issue.sprint_id = sprint.sprint_id
JOIN gros.sprint AS in_sprint
ON issue.project_id = in_sprint.project_id
AND in_sprint.sprint_id IN (...)
AND issue.updated <= CASE WHEN in_sprint.complete_date IS NOT NULL AND
    CAST(in_sprint.complete_date AS DATE) < CAST(in_sprint.end_date AS
    DATE) THEN in_sprint.complete_date ELSE in_sprint.end_date END
AND COALESCE(newer_issue.updated, CASE WHEN in_sprint.complete_date IS NOT
    NULL AND CAST(in_sprint.complete_date AS DATE) <
    CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date ELSE
    in_sprint.end_date END) >= CASE WHEN in_sprint.complete_date IS NOT
    NULL AND CAST(in_sprint.complete_date AS DATE) <
    CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date ELSE
    in_sprint.end_date END
WHERE issue.epic IS NOT NULL AND (sprint.sprint_id IS NULL OR
    COALESCE(CAST(sprint.start_date AS TIMESTAMP), CURRENT_TIMESTAMP()) >=
    CASE WHEN in_sprint.complete_date IS NOT NULL AND
    CAST(in_sprint.complete_date AS DATE) < CAST(in_sprint.end_date AS
    DATE) THEN in_sprint.complete_date ELSE in_sprint.end_date END) AND
    issue."type" = 7 AND COALESCE(issue.resolution, 0) NOT IN (1, 10000)
    AND COALESCE(issue.status, 0) NOT IN (6, 10008)
GROUP BY in_sprint.project_id, in_sprint.sprint_id, issue.epic
```

(b) Compiled

Figure B.10: Backlog epic points (refined query)

```
SELECT ${f(join_cols, "sprint", alias=T, sprint="in_sprint")},
    ${s(issue_key)} AS key, MAX(${s(story_points)}) AS story_points,
    MAX(${s(fix_version)}) AS fixversion
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS newer_issue
ON ${j(issue_next_changelog, "newer_issue", "issue")}
LEFT JOIN gros.${t("sprint")}
ON ${j(join_cols, "issue", "sprint")}
JOIN gros.${t("sprint")} AS in_sprint
ON ${j(join_cols, "issue", "in_sprint", 1)}
WHERE (${s(issue_open)} OR ${s(sprint_open)} >= ${s(sprint_open,
    sprint="in_sprint")})
AND ${s(issue_backlog)}
AND ${t("issue")}.updated <= ${s(sprint_open, sprint="in_sprint")}
AND (newer_issue.updated IS NULL OR newer_issue.updated > ${s(sprint_open,
    sprint="in_sprint")})
${g(join_cols, "sprint", f("issue_key"), sprint="in_sprint")}
```

(a) Template

```
SELECT in_sprint.project_id, in_sprint.sprint_id, issue.key AS key,
    MAX(CASE WHEN issue.story_points IN (-5, -1, 99, 100, 122, 999) THEN 0
    ELSE issue.story_points END) AS story_points, MAX(issue.fixversion) AS
    fixversion
FROM gros.issue
LEFT JOIN gros.issue AS newer_issue
ON newer_issue.issue_id = issue.issue_id AND newer_issue.changelog_id =
    issue.changelog_id + 1
LEFT JOIN gros.sprint
ON issue.project_id = sprint.project_id AND issue.sprint_id =
    sprint.sprint_id
JOIN gros.sprint AS in_sprint
ON issue.project_id = in_sprint.project_id
WHERE (issue.status NOT IN (5,6,10008) OR COALESCE(CAST(sprint.start_date
    AS TIMESTAMP), CURRENT_TIMESTAMP()) >=
    COALESCE(CAST(in_sprint.start_date AS TIMESTAMP), CURRENT_TIMESTAMP()))
AND issue."type" = 7 AND issue.story_points IS NOT NULL
AND issue.updated <= COALESCE(CAST(in_sprint.start_date AS TIMESTAMP),
    CURRENT_TIMESTAMP())
AND (newer_issue.updated IS NULL OR newer_issue.updated >
    COALESCE(CAST(in_sprint.start_date AS TIMESTAMP), CURRENT_TIMESTAMP()))
GROUP BY in_sprint.project_id, in_sprint.sprint_id, issue.issue_id,
    issue.key
```

(b) Compiled

Figure B.11: Backlog story points (original query)

```
SELECT ${f(join_cols, "sprint", alias=T, sprint="in_sprint")},
    ${s(issue_key)} AS key, MAX(${s(story_points)}) AS story_points,
    MAX(${s(fix_version)}) AS fixversion
FROM gros.${t("issue")}
LEFT JOIN gros.${t("issue")} AS newer_issue
ON ${j(issue_next_changelog, "newer_issue", "issue")}
LEFT JOIN gros.${t("sprint")}
ON ${j(join_cols, "issue", "sprint")}
JOIN gros.${t("sprint")} AS in_sprint
ON ${j(join_cols, "issue", "in_sprint", 1)}
AND in_sprint.sprint_id IN (${filter_sprint_ids})
AND ${t("issue")}.updated <= ${s(sprint_close, sprint="in_sprint")}
AND COALESCE(newer_issue.updated, ${s(sprint_close, sprint="in_sprint")})
    >= ${s(sprint_close, sprint="in_sprint")}
WHERE (${s(issue_open)} OR ${s(sprint_close)} >= ${s(sprint_close,
    sprint="in_sprint")}) AND ${s(issue_backlog)}
${g(join_cols, "sprint", f("issue_key"), sprint="in_sprint")}
```

(a) Template

```
SELECT in_sprint.project_id, in_sprint.sprint_id, issue.key AS key,
    MAX(CASE WHEN issue.story_points IN (-5, -1, 99, 100, 122, 999) THEN 0
    ELSE issue.story_points END) AS story_points, MAX(issue.fixversion) AS
    fixversion
FROM gros.issue
LEFT JOIN gros.issue AS newer_issue
ON newer_issue.issue_id = issue.issue_id AND newer_issue.changelog_id =
    issue.changelog_id + 1
LEFT JOIN gros.sprint
ON issue.project_id = sprint.project_id AND issue.sprint_id =
    sprint.sprint_id
JOIN gros.sprint AS in_sprint
ON issue.project_id = in_sprint.project_id
AND in_sprint.sprint_id IN (...)
AND issue.updated <= CASE WHEN in_sprint.complete_date IS NOT NULL AND
    CAST(in_sprint.complete_date AS DATE) < CAST(in_sprint.end_date AS
    DATE) THEN in_sprint.complete_date ELSE in_sprint.end_date END
AND COALESCE(newer_issue.updated, CASE WHEN in_sprint.complete_date IS NOT
    NULL AND CAST(in_sprint.complete_date AS DATE) <
    CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date ELSE
    in_sprint.end_date END) >= CASE WHEN in_sprint.complete_date IS NOT
    NULL AND CAST(in_sprint.complete_date AS DATE) <
    CAST(in_sprint.end_date AS DATE) THEN in_sprint.complete_date ELSE
    in_sprint.end_date END
WHERE (issue.status NOT IN (5,6,10008) OR CASE WHEN sprint.complete_date IS
    NOT NULL AND CAST(sprint.complete_date AS DATE) < CAST(sprint.end_date
    AS DATE) THEN sprint.complete_date ELSE sprint.end_date END >= CASE
    WHEN in_sprint.complete_date IS NOT NULL AND
    CAST(in_sprint.complete_date AS DATE) < CAST(in_sprint.end_date AS
    DATE) THEN in_sprint.complete_date ELSE in_sprint.end_date END) AND
    issue."type" = 7 AND issue.story_points IS NOT NULL
GROUP BY in_sprint.project_id, in_sprint.sprint_id, issue.issue_id,
    issue.key
```

(b) Compiled

Figure B.12: Backlog story points (refined query)