From Metrics to Metaphors: Emotional Software Visualisation for Interdisciplinary Understanding

Frieder Loch, Mirko Stocker, Simon Ott
Department of Computer Science
Eastern Switzerland University of Applied Sciences
Rapperswil, Switzerland
0000-0002-5206-7675, 0009-0002-2928-1646, 0009-0009-2287-5127

Sarah I. Hofer
Education and Educational Psychology
LMU Munich
Munich, Germany
0000-0001-7267-9356

***Abstract*—Software quality metrics are abstract and their implications are difficult to interpret, especially for non-technical stakeholders. Existing software visualisations do not bridge this gap and often require software development expertise. This paper proposes emotional software visualisation: an approach that maps code quality metrics (e.g., maintainability) to emotional metaphors grounded in everyday experiences (e.g., visual decay). We argue that such mappings can make software visualisations more intuitive for both technical and non-technical stakeholders as well as for novices and experienced developers. We illustrate the concept using a software city visualisation. A qualitative study with five participants suggested that the approach may support the understandability of quality metrics for novice and experienced developers. It also highlighted risks such as ambiguities in the interpretation of the metaphors. We conclude that emotional metaphors can augment existing software visualisations and provide a novel channel to convey quality metrics.**

***Index Terms*—Software visualisation, software quality metrics, emotional metaphors, emotional design, human-computer interaction, software city**

I. Introduction

Code quality describes whether source code satisfies non-functional requirements such as maintainability or security. Code quality metrics reduce positive or negative patterns to comparable values. However, these values hardly convey their implications for the software development process.

Quality metrics can be difficult to interpret. For example, the externally observable effects of low maintainability only manifest over time. To non-developers, the effects of quality degradation are only indirectly visible through slowing development or failures. Developers, however, are more aware of the state of the code and the urgency of present or future quality issues. Existing software visualisations fail to bridge the gap between continuous awareness by developers and failure-driven awareness by non-technical stakeholders.

Visualisations represent abstract data to amplify cognition [1]. However, most existing software visualisations represent code quality with numbers, colours, or graphs. The visualisations fail to convey how developers interpret the state of the code and the future implications of quality degradation. This is especially problematic for non-developers or novices struggling with the interpretation of these abstract metrics.

This paper proposes an emotional turn in software visualisation. We argue for the augmentation of software visualisations with metaphors that map quality metrics to real-world experiences to elicit an emotional response, and that such visualisations can support the interdisciplinary understanding of quality metrics.

We explore the following questions:

- How can code quality metrics be mapped to emotional metaphors in a structured way?
- How do users perceive emotional metaphors in comparison to conventional representations?
- How can emotional metaphors be integrated into existing software visualisations?

This paper contributes a conceptual framing of emotional software visualisation and a structured approach to map quality metrics and their implications to visual metaphors. The approach is illustrated using the software city visualisation [2]. The software city serves as the carrier for the emotional metaphors. We discuss the extension to other types of visualisations, such as graphs.

II. Related Work and Research Gap

Software visualisations aim to facilitate the understanding of computer software [3], [4] and typically display the structure, behaviour, and evolution of software [4], [5].

Visualisations often use metaphors to facilitate understanding through analogies. A frequently used metaphor to represent software systems is the city [2] or variations like the island metaphor [6]. These metaphors map the structure of a software system to the structure of a city. The original work of Wettel and Lanza focuses on the structure of a system and its classes by visualising the number of attributes and methods through the dimensions of the buildings [2].

Existing systems provide analytical visualisations using the dimensions of buildings or their colour and basic appearance. This limits their interpretability for non-experts. *VariMetrics* uses cracks in the building as an additional generic channel to visualise metrics [7]. The cracks are assigned to an arbitrary metric and are not derived from its implication. *Skylines* visualises structural properties of the software (e.g., the number of methods) through the shape of buildings [8]. Another approach sonifies the development activity through audio signals [9].

These systems indicate the feasibility of richer metaphors for software visualisation. However, the existing visualisations

apply these metaphors without emotional intent. They also lack a structured mapping process between metric and metaphor to explore the design space for emotional metaphors.

The existing work encodes information visually but does not leverage emotional metaphors that could provide a stronger sense of the state of a software project. While research suggests the importance of emotions in visualisation [10], the possibilities in software visualisation, especially regarding quality metrics, remain underexplored. A city visualisation, particularly in virtual reality (VR), positively affects emotions when compared with non-visual exploration in development environments [11]. However, this research uses the conventional city metaphor and measures emotions as an outcome instead of including it as a deliberate design consideration.

This paper addresses this gap with emotional metaphors that represent quality metrics and their implications across disciplines and levels of expertise.

III. Integration of Emotional Metaphors in Software Visualisation

We propose the integration of emotional metaphors into software visualisations. This paper illustrates this approach using the city metaphor but the method can be extended to other types of visualisations. We motivate this approach and provide a process to map quality metrics to emotional metaphors.

A. Motivation

Emotions can be located along valence (negative–positive) and arousal (calming–activating) [12]. This model provides the design space for the metaphors. Quality-degrading states are mapped to negative, activating regions (e.g., fear) to motivate attention and convey urgency while healthy states are mapped to positive, calming ones (e.g., comfort) to signal that no corrective action is required. Three considerations motivate the design of emotional metaphors.

- By linking emotions to specific material instead of inducing a general mood, **emotional design** can strengthen memory traces and hence support learning and understanding [13], [14]. Emotional metaphors connect to shared experiences, such as the gradual deterioration of a house. This approach creates a visualisation that builds on shared mental models rather than on learned conventions.
- The **dual-coding theory** [15] suggests that information that is processed through multiple channels is comprehended and retained more reliably. The visual metaphors engage a pictorial channel in addition to numerical representation of quality metrics.
- Software engineering has a long tradition of using **metaphors** to clarify abstract ideas, such as *inheritance* in object-oriented programming or the description of defects as *bugs*. One study found software-specific figurative expressions in roughly 9% of sentences in GitHub issues and pull requests [16]. Mapping quality metrics to emotionally grounded metaphors extends an existing practice and offers a shared vocabulary supporting the understandability for developers and non-developers alike.

B. Development of Emotional Metaphors

An emotional metaphor is a mapping from the implication of an abstract quality metric to a visual feature that evokes familiar real-world experiences and associated experiential responses. Conventional encodings, in contrast, map the metric value onto a visual property and must be learned as a convention. We explore whether emotional metaphors can support the interdisciplinary understanding of quality metrics and their implications. The development of the emotional metaphors follows a structured approach.

1) We define the **quality metric**. Common quality metrics that can be obtained from static code analysis are maintainability and security. We use the quality metrics of SonarQube [17]. Maintainability is based on the technical debt ratio, while security is rated by the severity of the most severe issue found. Thresholds for positive and negative states of the metric, and required levels in between (e.g., low, medium, high) are defined.
2) We define the **implication** of a poor or a good state of the metric on the software and the team (e.g., slower development).
3) We connect the implication to an **emotional response**. We identify emotional associations from the valence-arousal plane (e.g., fear, comfort) for positive and negative values of the metric.
4) We implement the emotional response as a **visual feature** (e.g., unease is evoked by a cracked building). This step provides the grounding in real-world experiences. The visual realisations should be validated to identify ambiguities in the mapping.

Table I shows the implementation of the outlined approach and develops exemplary mappings for two metrics (maintainability, security). Three of the realisations are discussed below.

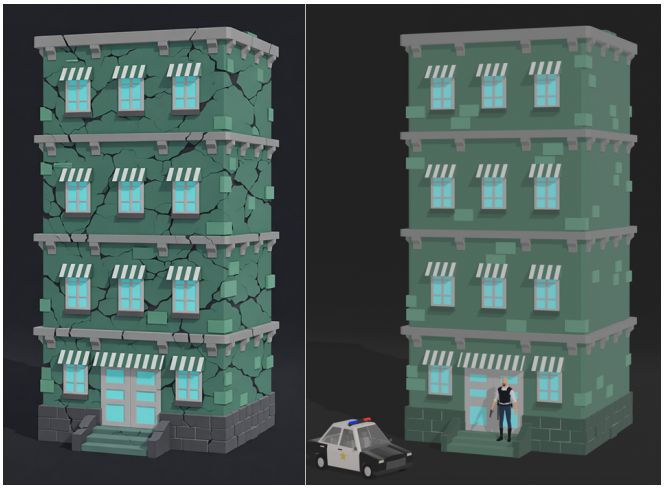
TABLE I: Mapping of Quality Metrics. Arrows mark the metric value direction (Maint. = Maintainability).

Metric	Implication	Emotion	Visual Feature
Maint. ↓	Slow change	Unease	Cracks, broken windows
Maint. ↑	Easy change	Comfort	Intact façade, tidy garden
Security ↓	Breaches	Fear	Police officer with car
Security ↑	Protection	Reassurance	Fortified door

Figure 1a visualises maintainability. A poor value leads to slow and difficult change. This implication should be related to the emotion of unease. This emotion is realised through the visual features of cracks and broken windows in the building.

Figure 1b shows an implementation of the metric security. A low value raises the possibility of security breaches. This should be connected to the emotion of fear. The realisation as a police officer indicates the possible ambiguity of emotional metaphors. Some people associate the presence of police with high security (crime has been prevented) while others may associate it with the opposite (crime is happening). Semantic and

cultural ambiguity has to be navigated to develop metaphors that are intuitive but also widely understandable.



(a) Low maintainability affects the outer appearance of the building.

(b) This security visualisation exemplifies a possible ambiguity.

Fig. 1: Visualisation of maintainability and security metrics.

If no broadly understandable metaphors can be identified, the ambiguities in emotional metaphors as exemplified by Figure 1b need to be mitigated. This can be done in line with the dual-coding theory by including redundant channels, such as colours or numerical values. Furthermore, additional textual explanations could support learning the metaphors.



Fig. 2: A tidy garden and an intact façade indicate high maintainability.

The absence of a negative metaphor does not convey good quality, but only the absence of information. Positive states therefore require their own metaphors that make upkeep explicit, as exemplified in Fig. 2. Such metaphors also provide the reference against which deterioration becomes legible.

IV. Qualitative Study

We conducted a qualitative interview study to explore how users perceive and interpret the emotional software visualisations and provide initial ideas and open research questions.

A. Setup of the Study

Five participants took part in the study: three novices (interns, $\leq$1 year of software development experience, no formal education in computer science) and two experienced developers (>5 years experience). Therefore, the study addressed only initial interpretation by novices and experienced developers and cannot provide insights into the perception of the metaphors by persons without any technical experience. Furthermore, we only targeted first impressions and not the correct interpretation of the metrics. The responses were recorded as written notes and thematically grouped (quotes were translated from German by the authors).

First, participants were introduced to the relevant quality metrics using a standardised description. They were then presented with several software visualisations. As baselines, these comprised a traditional quality dashboard (SonarQube [17]) and the software city metaphor visualisation (CodeCharta [**?**]). Subsequently, participants were shown six individual buildings that encoded the given quality metrics through emotional metaphors, including Fig. 1a and 1b. The evaluation concluded with a complete city composed of multiple buildings employing different metaphors. The buildings were authored to represent defined metric levels rather than generated from an analysed code base. The order was fixed so that participants first encountered conventional representations as a baseline before being introduced to the unfamiliar emotional metaphors.

We asked the participants four questions with regard to each visualisation:

- What is your first impression of the visualisation?
- How would you judge the quality of the system based on the visualisation?
- What do you feel when confronted with the visualisation?
- What actions would you suggest to improve the quality of the system?

At the end of the interview, the participants were asked to compare emotional and non-emotional metaphors and suggest possible applications in software development.

B. Results

The three novices reported difficulties interpreting numerical metrics and chart-based dashboard visualisations. In particular, they were uncertain whether a displayed value represented a good or bad state and which values should be considered acceptable. Colour encodings were identified as the most helpful visual cue for interpreting the dashboard.

Participants decoded the conventional city structurally without difficulty, mapping buildings to files. Quality, however, was inferred almost exclusively from the red-green encoding: four participants remained unsure which metric the building dimensions represented, even though the mapping was shown in the interface.

The emotional metaphors were perceived as more immediate than both the dashboard and the conventional city representation. However, some metaphors were interpreted differently or associated with the wrong quality metric, indicating ambiguities in the mapping. One participant described the cracked

building in Fig. 1a as "not changed in a long time", capturing neglect but not maintainability itself. Visual representations associated with severe problems, such as structural decay, were perceived as requiring immediate attention.

Three participants considered the emotional visualisations more engaging than both baseline representations. The experienced participants saw the visualisations as a communication tool for non-technical stakeholders, enabling a quick understanding of the overall project state. At the same time, all participants expressed the need for quantitative information to support software quality improvement activities and identify concrete next steps. Participants mentioned a risk of oversimplification and misunderstanding when relying solely on emotional metaphors. Confronted with the complete city, one participant asked which elements carried meaning and which did not, and anticipated wrong conclusions.

These observations are limited by the small sample, the absence of non-technical participants, and the use of self-reports instead of standardised instruments to measure emotional impact. The findings therefore indicate directions for further exploration rather than evidence of an effect.

V. Discussion and Future Work

This paper argues for emotional metaphors in software visualisation. This idea was exemplified using the city metaphor but can be generalised to other visualisations. The extension of emotional metaphors requires interdisciplinary collaboration involving human-computer interaction, psychology, and empirical software engineering.

Emotional metaphors carry risks. Their ambiguity is a challenge for their practical application, as observed in the study. They are more exposed to misinterpretations than traditional metaphors that rely on conventions, such as colour coding. Redundant encoding of information on different channels, for instance with numerical values, may address this issue. Emotional metaphors might oversimplify complex software engineering matters and direct attention to symptoms rather than root causes. A building with cracks shows that something is wrong but not why. Displaying cracks in a building may overdramatise the situation, leading to alarm fatigue. Strong imagery may also evoke unrelated associations, for instance when structural collapse recalls experiences of disaster. The design should consider an intensity ceiling. Visual features should remain within the range of deterioration and avoid depictions of catastrophe or harm.

The emotional response may depend on the feeling of presence that describes whether users have a feeling of being in the virtual environment and not in the physical world. Higher levels of presence can increase engagement and support task performance and learning [18]. Presence is supported by the plausibility of the environment. Research suggests that meaning-making is grounded in embodied interaction, where understanding emerges through situated, physical engagement rather than abstract representations [19]. Interaction with the environment (e.g., by being able to enter a decaying building) could strengthen the emotional response.

A. Further Evaluation and Development

The hypothesis that there are broadly shared interpretations of metaphors needs evaluation. The interpretation might be culturally specific or influenced by previous knowledge. Therefore, methodical evaluations are required.

First, the mapping between metaphors and metrics should be established. To this end, participants with different backgrounds could rate to what degree a certain visualisation is associated with a certain quality metric. This stage should also validate whether the visualisations induce an emotional reaction, using a standardised instrument for valence and arousal. It should also be addressed whether different levels of a metric (e.g., high, medium, low) can be distinguished.

Second, the metaphors should be implemented in a visualisation system. This system can be used to measure how the visualisation influences the effectiveness of code review (e.g., the impact on code review coverage or participation) or the awareness of the state of quality by non-technical stakeholders.

B. Exploring Composed Metaphors

This paper focused on the discussion of individual metrics. However, visualising several metrics is critical for a complete software visualisation. For instance, a metaphor that combines low maintainability and high security would be required. The design of such metaphors should ensure emotional content while not sacrificing intuitiveness.

Composed metaphors can be created by combining distinct visual features, each mapped to a different metric. The case mentioned above (high security and low maintainability) could be rendered as a fortified door (security) in a decaying building (maintainability). Both metrics remain separately legible but are integrated in one building. It should be evaluated whether users can decode the components separately and whether the composed metaphor still produces an emotional response.

C. Integration into Software Development

Visualisations support a shared understanding in software development, since stakeholder priorities differ and refactoring "benefits are typically long-term, and may not be immediately visible" [20]. Visualisations highlight areas requiring additional review and refactoring effort. A crumbling building might create a sense of urgency among all stakeholders.

Future work should investigate the application of emotional metaphors in code reviews and team retrospectives. Software visualisation can also represent test-related metrics such as code coverage [21]. Evaluations may assess whether the emotional metaphors influence decision-making processes or prioritisation of refactoring over other activities. These measurements should also address qualitative aspects, for instance the effect of the visualisation on team processes.

D. Beyond the City Metaphor

The idea of emotional metaphors can be transferred to further types of software visualisations, for instance graphs. Evaluations should address whether the benefits of emotional metaphors also manifest in non-immersive visualisations.

Graph-based visualisations depict the relationships between software entities. Figure 3 shows an example where each node carries emotional metaphors, such as vegetation. The represented metrics extend beyond static code analysis. Thriving plants indicate healthy components, while wilting vegetation signals gradual quality degradation. Architectural decay marks components in a critical state, and weather conditions represent external factors such as failure rates or load. Problematic dependency structures may additionally appear clustered or tangled, showing unclear relations or circular dependencies.

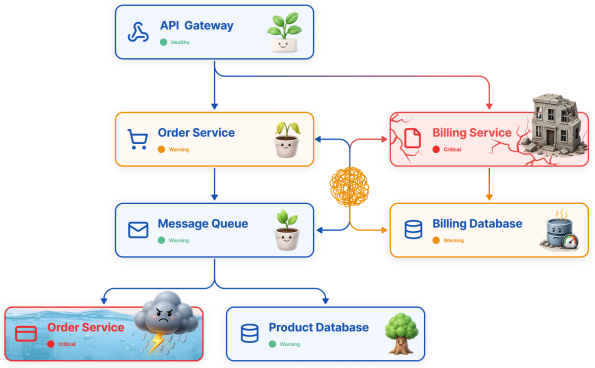


Fig. 3: A graph-based visualisation introducing emotional metaphors to show quality issues.

VI. Conclusion

This paper proposes the augmentation of existing analytical software visualisations with emotional metaphors grounded in real-world experiences. It presented a structured process that maps a quality metric through its implication and an associated emotional response to a visual feature. A qualitative study indicated that such metaphors are perceived as more immediate and engaging than conventional representations, but that individual mappings remained ambiguous.

Several research questions remain, for instance how ambiguous metaphors can be mitigated, how several metrics can be combined in one visual feature, and how these metaphors can be applied in software development. Exploring these ideas invites interdisciplinary collaboration across human-computer interaction, psychology, and empirical software engineering.

References

[1] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in information visualization: using vision to think*. San Francisco, Calif.: Morgan Kaufmann, 2007.

[2] R. Wettel and M. Lanza, "Visualizing Software Systems as Cities," in *2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*. IEEE, Jun. 2007, pp. 92–99. [Online]. Available: https://ieeexplore.ieee.org/document/4290706

[3] B. A. Price, R. M. Baecker, and I. S. Small, "A Principled Taxonomy of Software Visualization," *Journal of Visual Languages & Computing*, vol. 4, no. 3, pp. 211–266, Sep. 1993. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1045926X83710153

[4] A.-L. Mattila, P. Ihantola, T. Kilamo, A. Luoto, M. Nurminen, and H. Väätäjä, "Software visualization today: systematic literature review," in *Proceedings of the 20th International Academic Mindtrek Conference*. Tampere Finland: ACM, Oct. 2016, pp. 262–271. [Online]. Available: https://dl.acm.org/doi/10.1145/2994310.2994327

[5] A. Kuhn and M. Stocker, "CodeTimeline: Storytelling with versioning data," in *2012 34th International Conference on Software Engineering (ICSE)*, Jun. 2012, pp. 1333–1336. [Online]. Available: https://ieeexplore.ieee.org/document/6227086

[6] M. Misiak, A. Schreiber, A. Fuhrmann, S. Zur, D. Seider, and L. Nafeie, "IslandViz: A Tool for Visualizing Modular Software Systems in Virtual Reality," in *2018 IEEE Working Conference on Software Visualization (VISSOFT)*, Sep. 2018, pp. 112–116. [Online]. Available: https://ieeexplore.ieee.org/document/8530137

[7] J. Mortara, P. Collet, and A.-M. Pinna-Dery, "Customizable visualization of quality metrics for object-oriented variability implementations," in *Proceedings of the 26th ACM International Systems and Software Product Line Conference - Volume A*. Graz Austria: ACM, Sep. 2022, pp. 43–54. [Online]. Available: https://dl.acm.org/doi/10.1145/3546932.3547073

[8] M. Giannaccari, M. Raglianti, and M. Lanza, "Skylines: Visualizing Object-Oriented Software Systems Through Class Contours," in *2025 IEEE Working Conference on Software Visualization (VISSOFT)*, Sep. 2025, pp. 64–68, iSSN: 2832-6555. [Online]. Available: https://ieeexplore.ieee.org/document/11175654

[9] C. Armenti, M. Raglianti, and M. Lanza, "Sonifying and Visualizing the Heartbeat of Evolving Software Systems," in *2025 IEEE Working Conference on Software Visualization (VISSOFT)*, Sep. 2025, pp. 1–12. [Online]. Available: https://ieeexplore.ieee.org/document/11175666

[10] X. Lan, Y. Wu, and N. Cao, "Affective Visualization Design: Leveraging the Emotional Impact of Data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 1, pp. 1–11, Jan. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10301796/

[11] S. Romano, N. Capece, U. Erra, G. Scanniello, and M. Lanza, "The city metaphor in software visualization: feelings, emotions, and thinking," *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 33 113–33 149, Dec. 2019. [Online]. Available: http://link.springer.com/10.1007/s11042-019-07748-1

[12] J. A. Russell, "A circumplex model of affect." *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, Dec. 1980. [Online]. Available: https://doi.apa.org/doi/10.1037/h0077714

[13] R. M. Wong and O. O. Adesope, "Meta-Analysis of Emotional Designs in Multimedia Learning: A Replication and Extension Study," *Educational Psychology Review*, vol. 33, no. 2, pp. 357–385, Jun. 2021. [Online]. Available: https://doi.org/10.1007/s10648-020-09545-x

[14] J. L. Plass and U. Kaplan, "Emotional Design in Digital Media for Learning," in *Emotions, Technology, Design, and Learning*. Elsevier, 2016, pp. 131–161. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128018569000074

[15] A. Paivio, *Mental Representations*. Oxford University Press, Sep. 1990. [Online]. Available: https://academic.oup.com/book/10932

[16] M. M. Imran, P. Chatterjee, and K. Damevski, "Shedding Light on Software Engineering-specific Metaphors and Idioms," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. Lisbon Portugal: ACM, Apr. 2024, pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3597503.3639585

[17] SonarSource, "Understanding measures and metrics," 2026. [Online]. Available: https://docs.sonarsource.com/sonarqube-server/user-guide/code-metrics/metrics-definition

[18] G. Makransky and G. B. Petersen, "The Cognitive Affective Model of Immersive Learning (CAMIL): a Theoretical Research-Based Model of Learning in Immersive Virtual Reality," *Educational Psychology Review*, vol. 33, no. 3, pp. 937–958, Sep. 2021. [Online]. Available: https://link.springer.com/10.1007/s10648-020-09586-2

[19] P. Dourish, *Where the Action Is: The Foundations of Embodied Interaction*. Cambridge: MIT Press, 2001.

[20] J. Chen, J. Xiao, Q. Wang, L. J. Osterweil, and M. Li, "Perspectives on refactoring planning and practice: an empirical study," *Empirical Software Engineering*, vol. 21, no. 3, pp. 1397–1436, Jun. 2016. [Online]. Available: https://doi.org/10.1007/s10664-015-9390-8

[21] G. Balogh, T. Gergely, A. Beszedes, and T. Gyimothy, "Using the City Metaphor for Visualizing Test-Related Metrics," in *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*. Suita, Osaka, Japan: IEEE, Mar. 2016, pp. 17–20. [Online]. Available: http://ieeexplore.ieee.org/document/7476755/