Undo and Redo in Concurrent Multifaceted Low-code Application Models

HUGO LOURENÇO, OutSystems, Portugal
CARLA FERREIRA, NOVA LINCS, NOVA School of Science and Technology, Portugal
JOÃO COSTA SECO, NOVA LINCS, NOVA School of Science and Technology, Portugal

Low-code platforms follow a model-driven development approach to manipulate code at a high level of abstraction (*cf.* DSLs [9], visual and block-based [10], reactive and dataflow [1, 6–8], event-driven [11, 12, 21, 25]) and support advanced editing capabilities such as visual languages [16, 19], code templates [14, 15, 22], and AI assistance [17, 20]. Application models are structured documents often represented in dynamic data structures and directly accessed by editors [2, 18]. As a result, the interaction with these models is richer than the interactions with text files in traditional programming languages. For instance, in the OutSystems platform, an application is a well-formed directed graph of objects edited by a desktop-based IDE that runs several processes simultaneously to check and modify the model [18].

OutSystems is moving from a desktop-based IDE, where developers collaborate by editing models locally and subsequently merging changes, to a live, web-based, AI-assisted environment [17], with collaborative editing and offline capabilities on the horizon. AI-based helpers and web-based editors introduce concurrent (asynchronous) and speculative edits (suggestions) that require a new design paradigm. Editing operations may consist of incomplete intermediate states, but can be committed only if the resulting model is well-formed. The intermediate states must not prevent other developers from editing, compiling, and testing the application. Multifaceted models (*cf.* [5, 13]) allow for multiple isolated views, enabling the compiler and AI-based assistants to work in the background over a sound model while it is being edited, and the IDE to preview multiple AI-generated suggestions without changing the actual model. The model also acts as a guardrail for AI generative procedures - AI-generated models must be checked automatically for soundness before being presented to the developer. To address these challenges, we propose a transactional framework with three kinds of transactions: write transactions for the regular user edits of the model; read-only transactions for tasks that scan the model in the background and need a stable view (e.g. AI engines and the compiler); and discardable transactions to speculatively produce non-persistent alternative versions of the model.

Our work is based on the mechanism of versioned boxes used in software transactional memory [3], and conflict-free replicated data types [23] to implement multifaceted models [4, 5]. We use versioned boxes to implement *Undo Redo* and to quickly identify the modified parts of a model, minimising the compiler's work. Naïve use of versioned boxes requires class fields to be replaced by versioned boxes (e.g. using `VersionedBox<int>` instead of `int`), increasing the memory footprint. We use a hybrid approach: fields keep their original type and are used as the default storage for unmodified values. Only modified fields need a versioned box, which is stored externally to the object. Compared to naïve versioned boxes, our approach delivers up to 30% memory usage decrease. Memory consumption overhead starts at zero for a freshly loaded model and increases proportionally to the number of edits (up to the *Undo Redo* history limit). In the worst case where all fields of all objects are modified, it is equivalent to using plain versioned boxes.

We are applying out framework to MORPHEUS [17], which is a cloud-based IDE. Efficient memory usage is important to keep running costs low by loading as many models as possible in a single server instance. This work is a first step towards a low-code model editor with offline capabilities, where different versions represent editing threads and collaborative editing with *Undo Redo* [24].

Authors' addresses: Hugo Lourenço, OutSystems, Linda-a-Velha, Portugal, hugo.lourenco@outsystems.com; Carla Ferreira, NOVA LINCS, NOVA School of Science and Technology, Caparica, Portugal, carla.ferreira@fct.unl.pt; João Costa Seco, NOVA LINCS, NOVA School of Science and Technology, Caparica, Portugal, joao.seco@fct.unl.pt.

REFERENCES

[1] Umut A. Acar, Guy E. Blelloch, and Robert Harper. Adaptive functional programming. In *Proceedings of the 29th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '02, page 247–259, New York, NY, USA, 2002. Association for Computing Machinery.

[2] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. *Model-driven software engineering in practice*. Morgan & Claypool Publishers, 2017.

[3] João P. Cachopo and António Rito Silva. Versioned boxes as the basis for memory transactions. *Sci. Comput. Program.*, 63(2):172–185, 2006.

[4] João Campinhos, João Costa Seco, and Jácome Cunha. Type-safe evolution of web services. In *2017 IEEE/ACM 2nd International Workshop on Variability and Complexity in Software Design (VACE)*, pages 20–26, 2017.

[5] Luis Carvalho and João Costa Seco. Deep semantic versioning for evolution and variability. In Niccolò Veltri, Nick Benton, and Silvia Ghilezan, editors, *PPDP 2021: 23rd International Symposium on Principles and Practice of Declarative Programming, Tallinn, Estonia, September 6-8, 2021*, pages 21:1–21:13. ACM, 2021.

[6] Gregory Harold Cooper. *Integrating dataflow evaluation into a practical higher-order call-by-value language*. PhD thesis, Brown UniversityDepartment, 2008.

[7] Evan Czaplicki and Stephen Chong. Asynchronous functional reactive programming for guis. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '13, page 411–422, New York, NY, USA, 2013. Association for Computing Machinery.

[8] Miguel Domingues and João Costa Seco. Type safe evolution of live systems. In *Workshop on Reactive and Event-based Languages & Systems (REBLS'15)*, 2015.

[9] Martin Fowler. *Domain-Specific Languages*. The Addison-Wesley signature series. Addison-Wesley, 2011.

[10] Neil Fraser, Quynh Neutron, Ellen Spertus, and Mark Friedman. Blockly, 2014. https://developers.google.com/blockly, Accessed: 2024-06-28.

[11] Leandro Galrinho, João Costa Seco, Søren Debois, Thomas T. Hildebrandt, Håkon Normann, and Tijs Slaats. Regrada: Reactive graph data. In Ferruccio Damiani and Ornela Dardha, editors, *Coordination Models and Languages - 23rd IFIP WG 6.1 International Conference, COORDINATION 2021, Held as Part of the 16th International Federated Conference on Distributed Computing Techniques, DisCoTec 2021, Valletta, Malta, June 14-18, 2021, Proceedings*, volume 12717 of *Lecture Notes in Computer Science*, pages 188–205. Springer, 2021.

[12] Thomas T. Hildebrandt and Raghava Rao Mukkamala. Declarative event-based workflow as distributed dynamic condition response graphs. In Kohei Honda and Alan Mycroft, editors, *Proceedings Third Workshop on Programming Language Approaches to Concurrency and communication-cEntric Software, PLACES 2010, Paphos, Cyprus, 21st March 2010*, volume 69 of *EPTCS*, pages 59–73, 2010.

[13] Piotr Kaminski, Marin Litoiu, and Hausi Müller. A design technique for evolving web services. In *Proceedings of the 2006 Conference of the Center for Advanced Studies on Collaborative Research*, CASCON '06, page 23–es, USA, 2006. IBM Corp.

[14] Hugo Lourenço, Carla Ferreira, and João Costa Seco. OSTRICH - A type-safe template language for low-code development. In *24th International Conference on Model Driven Engineering Languages and Systems, MODELS 2021, Fukuoka, Japan, October 10-15, 2021*, pages 216–226. IEEE, 2021.

[15] Hugo Lourenço, Carla Ferreira, João Costa Seco, and Joana Parreira. OSTRICH: a rich template language for low-code development (extended version). *Softw. Syst. Model.*, 22(5):1645–1663, 2023.

[16] Hugo Lourenço, Joana Tavares, Rui Eugénio, Miguel Lourenço, and Tiago Simões. LUV is not the answer: continuous delivery of a model driven development platform. *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, 2020.

[17] OutSystems. Project MORPHEUS - generative ai, 2023. https://www.outsystems.com/news/generative-ai-roadmap/.

[18] OutSystems. Platform overview. https://www.outsystems.com/platform/, 2024.

[19] João Ramalho, Hugo Lourenço, and João Costa Seco. From builders to editors: Bidirectional transformations of low-code models. In *ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion, MODELS 2021 Companion, Fukuoka, Japan, October 10-15, 2021*, pages 108–117. IEEE, 2021.

[20] João Costa Seco, Jonathan Aldrich, Luis Carvalho, Bernardo Toninho, and Carla Ferreira. Derivations with holes for concept-based program synthesis. In Christophe Scholliers and Jeremy Singer, editors, *Proceedings of the 2022 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, Onward! 2022, Auckland, New Zealand, December 8-10, 2022*, pages 63–79. ACM, 2022.

[21] João Costa Seco, Søren Debois, Thomas T. Hildebrandt, and Tijs Slaats. RESEDA: declaring live event-driven computations as reactive semi-structured data. In *22nd IEEE International Enterprise Distributed Object Computing Conference, EDOC 2018, Stockholm, Sweden, October 16-19, 2018*, pages 75–84. IEEE Computer Society, 2018.

[22] João Costa Seco, Hugo Lourenço, Joana Parreira, and Carla Ferreira. Nested OSTRICH: hatching compositions of low-code templates. In Eugene Syriani, Houari A. Sahraoui, Nelly Bencomo, and Manuel Wimmer, editors, *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems, MODELS 2022, Montreal, Quebec, Canada, October 23-28, 2022*, pages 210–220. ACM, 2022.

[23] Marc Shapiro, Nuno M. Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free replicated data types. In Xavier Défago, Franck Petit, and Vincent Villain, editors, *Stabilization, Safety, and Security of Distributed Systems - 13th International Symposium, SSS 2011, Grenoble, France, October 10-12, 2011. Proceedings*, volume 6976 of *Lecture Notes in Computer Science*, pages 386–400. Springer, 2011.

[24] Leo Stewen and Martin Kleppmann. Undo and redo support for replicated registers. In *Proceedings of the 11th Workshop on Principles and Practice of Consistency for Distributed Data, PaPoC 2024, Athens, Greece, 22 April 2024*, pages 1–7. ACM, 2024.

[25] Glynn Winskel. An introduction to event structures. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, School/Workshop*, page 364–397, Berlin, Heidelberg, 1988. Springer-Verlag.