The Database Version Approach: Overview and Potential Directions

In tribute to Geneviève Jomier (1948 - 2018)

Talel Abdessalem
LTCI, Télécom ParisTech, Université Paris-Saclay
Paris, France
talel.abdessalem@telecom-paristech.fr

Claudia Bauzer Medeiros
University of Campinas, Institute of Computing (IC - UNICAMP)
Campinas, Brazil
cmbm@ic.unicamp.br

Wojciech Cellary
Poznan University of Economics
Poznan, Poland
cellary@kti.ue.poznan.pl

Stéphane Gançarski
LIP6 - U.P.M.C
Paris, France
Stephane.Gancarski@lip6.fr

Khaled Jouini
MARS Research Lab LR17ES05, ISITCom, University of Sousse
Sousse, Tunisia
khaled.jouini@isitc.u-sousse.tn

Maude Manouvrier
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE
Paris, France
maude.manouvrier@dauphine.fr

Marta Rukoz
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE
Paris, France
marta.rukoz@dauphine.fr

Michel Zam
Université Paris-Dauphine, PSL Research University, CNRS UMR [7243] LAMSADE
Paris, France
zam@dauphine.fr

ABSTRACT

In 1990, W. Cellary and G. Jomier proposed the Database Version (DBV) approach, which allows to manage multiversion databases – those in which several versions of a set of data items coexist. Ever since, its model, theory and algorithms have been adopted in a multitude of research initiatives and publications, and been applied to a variety of applications, in particular those in which there is a need for keeping track of parallel or (spatio)-temporal evolution of states of the world. This article presents an overview of the DBV approach, and some of the associated research initiatives throughout three decades, pointing out new potential directions. It has been written in tribute to Geneviève Jomier, Prof. Emeritus of The Université of Paris-Dauphine, who left us in March 2018.

1 INTRODUCTION

This article presents the *Database Versions* (DBV) approach, proposed by W. Cellary and G. Jomier in 1990 in [10].

Formally, a database is monoversion, representing only one state of the modeled world; each data item has a single value. In the DBV approach, a multiversion database brings together several states of the world, these states being variants or evolutions in time of the modeled world. Each data item has, in this case, several versions, the value of that item potentially changing from one version to another. Each Database Version represents a consistent state or a configuration of the modeled world [17]. DBVs can be applied to any type of data, whether images [35, 36], documents [5] or spatio-temporal data [42, 43, 48].

As will be seen throughout this text, the DBV model can be adopted in a variety of situations, whenever there is a need for managing many states of the world. Though this may seem to be an obvious statement, since it models versions (and thus states of the world), it is simple and generic. Through the DBV model, versions can be managed through a small set of compact data structures, simplifying maintenance of co-existing states, and speeding up rollback to any database state. For this reason, it was implemented in many computing platforms, for several purposes, in a variety of contexts.

The DBV model has originated several research initiatives, including the ones presented in a variety of papers - e.g., [2, 5, 17, 24, 32, 33, 36–39, 48, 49, 52]. Several PhD thesis in computer science [1, 18, 22, 35, 42, 50], supervised by G. Jomier, are based on this model. The model has also inspired thesis [41] and research projects - e.g., [13, 31, 44], in other institutions.

Section 2 presents the DBV approach and Section 3 discusses a few of its applications. Section 4 concludes the paper, presenting new potential research directions which can still be exploited.

2 DBV APPROACH

This section gives an overview of the DBV approach and of its subsequent generalizations.

2.1 Overview of the concepts

A DBV is identified by v. A multiversion data is associated with an immutable identifier, d. A DBV contains exactly one *logical version* d_j of each multiversion data d, having an identifier and a value val.

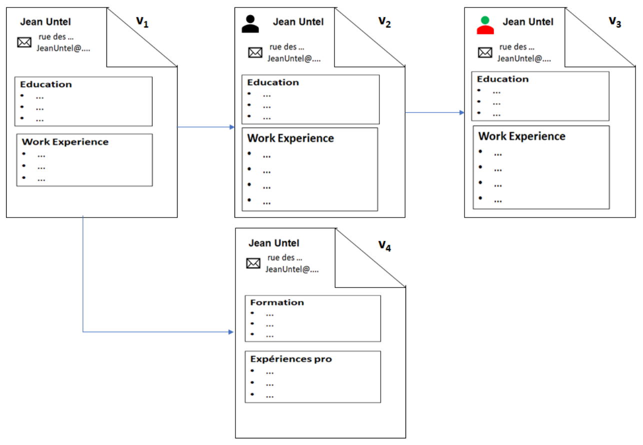

Figure 1: An example of a multiversion curriculum vitae (from [3]).

The identifier of the logical version of the data d_j in a VBD v_i is the pair (d_j, v_i).

A DBV is derived from an existing one, by logical copy, links between DBVs being stored in a *derivation tree*.

Figure 1 shows a simple example of a multiversion database, each DBV modeling a curriculum vitae (CV). The multiversion database contains 4 DBV, identified by v_1 to v_4, each one corresponding to a version of the CV. DBV v_2 has been created from DBV v_1, by adding a B&W picture and by enlarging the 'Work Experience' section. It corresponds to a long version of the CV, while DBV v_1 is a short one. DBV v_3 has been created from v_2, replacing the picture by a colored one. DBV v_4 is a French version of the CV.

A multiversion database has two levels: the logical level (what the user sees) and the physical one (what is actually stored). At physical level, logical versions, which share the same value *val* in several DBVs, share the same *physical version*, containing value *val*. An *association table* contains the link between each logical version (d, v) and the associated physical version p which contains value *val*. When a data does not exist in a DBV, its value is $\perp$, meaning "does not exist".

Figure 2 represents the physical level of the multiversion CV database of Figure 1. The association table of section 'Education' of the CV shows that the physical version, *explicitly* associated with DBV v_1, is *implicitly* associated with DBV v_2 and v_3. On the other hand, the French version is explicitly associated with DBV v_4. The association table of the picture shows that DBV v_1 does not contains any picture (the value is $\perp$), DBV v_2 contains a B&W picture while DBV v_3 contains a colored one.

2.2 Generalization of the DBV Model

The original article [10] presents the mechanisms for version identification, data updates, concurrency and data consistency management, as well as management of complex objects. The model was subsequently generalized: formalization and manipulation tools are presented in [16–18], an integrity constraint mechanism to maintain consistency in multiversion databases has been proposed

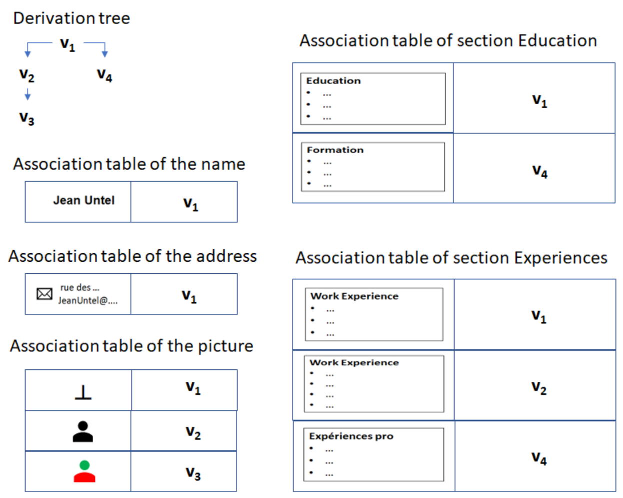
Derivation tree
V1
V2
V3
V4
Association table of the name
Jean Untel
Association table of the address
rue des ...
JeanUntel@....
Association table of the picture
⊥
Association table of section Education
Education
Formation
Association table of section Experiences
Work Experience
Expériences pro

Figure 2: The physical level of the multiversion CV database of Figure 1 (from [3]).

in [14], a query language is defined in [1, 2], as well as indexing mechanisms in [24] and storage strategies in [22].

3 DOMAIN APPLICATIONS

The DBV model is generic and has been applied to several types of data in many domains. This section briefly describes some of the main results.

3.1 Software life-cycle traceability

Karma [50] is an extension of the DBV model that adds execution traces to data versions. MyDraft [52], an award winning agile online platform meant to design and run web apps in minutes, is a practical implementation of Karma for the agile software lifecycle process and was successfully used in dozens of both professional and academic projects [9, 51].

MyDraft manages co-evolution of executable objects, models and meta-models by making them live together in the same cloud-based environment, and simultaneously available to a huge number of end-users, designers and programmers [52]. Design time and runtime activities are collapsed in a storage space-time continuum, managed by a multiversion database. Full traceability features — including automatic versioning, unlimited undo-redo, time machine and advanced consistency mechanisms based on execution traces — provide fast feedback through a continuous delivery loop, therefore helping scientists, inovators, designers and business analysts quickly explore new ideas and select the best choices faster.

3.2 Spatio-temporal and multifocus data management

In [33, 42, 43, 48], the DBV model has been used to manage spatio-temporal data, for which it is necessary to offer multi-scale management mechanisms, an object being able to appear or disappear, be aggregated or not, depending on the scale (spatial or temporal) of the data. [42, 43] use DBV in the context of urban evolution, whereas [31, 33] adopt the model for the management of environmental resources.

A good example of the need to handle scale diversity is that of climate change research. It requires collaboration of scientists that study the world at multiple space and time scales, and from distinct perspectives. Studies (and research focus) vary from the large (world climate) to the small (microclimate) to the smaller (e.g., effects of micro-organisms on the environment). This, in turn, results in a highly heterogeneous research scenario that must cope with a wide variety of data sources, collecting devices, methodologies, models and visualization needs, and hence huge interoperability challenges.

Moreover, the notion of multiple scales goes beyond observable phenomena (e.g., in climate change, in urban evolution, or in molecular biology). In particular, the work of [48] takes advantage of the DBV model to study what they call a "focus", and scenarios posed by "multi-focus" research. A focus, here, is a perspective from which a given problem can be analyzed, similar to the notion of a database view. Unlike views, it cannot be modelled through a database query. It encompasses not only the data, but constraints applicable to that specific focus. The research shows that, by supporting multiple foci (using DBV), one can build a collaborative research environment, in which groups of scientists can work on distinct aspects of a given scenario.

3.3 Bitemporal databases

In [15], author presents a new approach to implement an object bitemporal database where both valid-time (the time of a fact in the real world) and transaction-time (the time a fact is recorded in the database) are represented. This approach is based on the DataBase Version model: time is attached to possible states of the whole real world, represented by a DBV, which facilitates the manipulation of past events and allows a straightforward representation of branching evolution in valid-time. Moreover, as in the DBV model any existing version can be modied, it is possible to modify any past fact, which is necessary in a bitemporal database used to correct and keep traces of errors made on the history of real world entities. The paper [15] describes issues of the approach : time representation, manipulation primitives and alternatives, as well as the Modesty_2*T* prototype which implements it on top of the O2 OODBMS [7].

3.4 Storage and management of similar images

In [21, 35, 36], the DBV approach has been adapted to image management, particularly in the context of image processing. Image processing is an iterative trial and error process: in order to improve the image quality or the detection of the objects the image contains, several operations or series of operations can be applied to the whole image or to well chosen parts of it, generating new intermediate images as result. Throughout the process, intermediate images should be saved, allowing to start another sequence of operations, from an intermediate image, that could generate a better result. To minimize the memory space of a set of images and speed up several image processing operations, a structure, called *Generic Quadtree (GQT)*, has been proposed in [21, 36]. This structure is based on the DBV approach, an image corresponding, in this case, to a DBV and the processed parts of the image to the versionned

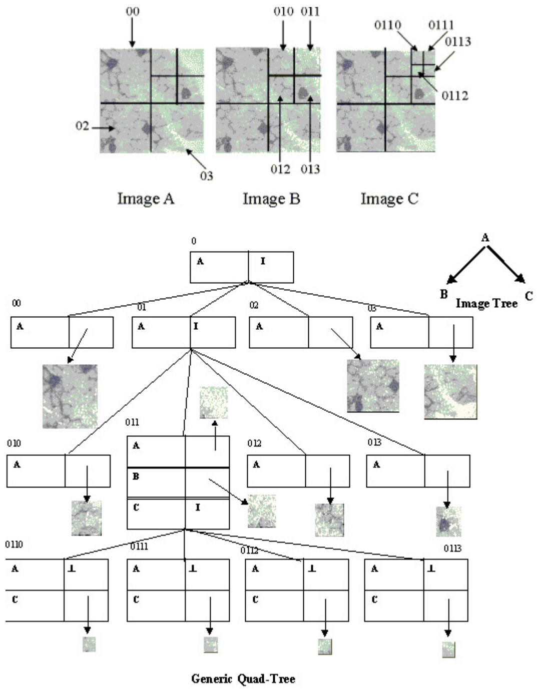

Figure 3: Three processed images stored in a GQT (from [21]).

data. A GQT stores a set of related images organized in quadtrees[1]. Figure 3 represents a example of a GQT storing three processed images. Image *A* is the original one. Image B results from darkening quadrant 001 of image *A* and Image *C* has been created from Image *A*, by dividing quadrant 001 and applying dilatation to the resulting sub-quadrants.

The nodes of a GQT are called *generic nodes*. For each quadrant appearing in an image, there is a node with the same identifier in the GQT. A generic node *n* represents quadrants *n* of all the stored images. The GQT is based on a principle of sharing of quadrants values among images, using a similarity distance between quadtrees, called *Q-similarity*. The Q-similarity between two images, *i* and *j*, is defined as the number of quadrants having the same identifier but different values in both images, divided by the cardinality of the set containing the image quadrant identifiers appearing in both images. In order to increase the implicit sharing of node values and consequently to reduce the size of generic nodes, an image *j* is inserted in the *Image Tree* (equivalent to the derivation tree of the DBV approach) as a descendant of the image *i* having the smallest *Q-similarity* with image *j*. In Figure 3 all image quadrants, except the processed quadrant 001 and its sub-quadrants, are shared between all images. Value *I* in the GQT means that a quadrant has been divided.
In fact, GQT is more than a storage structure. It is designed to be inserted in an environment for image management processing, because it allows users to extract one or several images easily and

[1]A quadtree is a well-known unbalanced spatial data structure built by recursive divisions of space in four equal-size disjoint quadrants - see in [47].

to work on them, delete or insert new images, compare images, build images sequences, etc. This approach has been extended to content-based image retrieval, defining multi-level index [20] and dissimilarity metrics between images [45].

3.5 Versions of uncertain Tree-Structured documents

In domains such as engineering and collaborative editing, version control has been always considered as a crucial task that enables maintaining the different versions of a shared object (e.g. software module or project report) and the history of the changes. This enables and eases the resolution of bugs, querying past versions, and the integration of content from various contributors. Much efforts related to version control [4, 30], in both research and applications, have been done, but mainly in contexts seemed as deterministic, that is in which participants are reliable and the shared content evolves without uncertainties. In this context, robust version control tools such as Subversion [12] and Git [11] properly manage large repositories of source codes and shared file systems.

Nowadays, there is an increasing interest in applications inherently uncertain using version control, such as web-scale collaborative platforms: Platforms such as Wikipedia[2] or Google Docs [3] enable unbounded interactions between a large number of contributors, without prior knowledge of their level of expertise and reliability. In these systems, version control is used for keeping track of the evolution of the shared content and its provenance. In such environments, uncertainty is ubiquitous due to the unreliability of the sources, the incompleteness and imprecision of the contributions, the possibility of malicious editing and vandalism acts, etc. Therefore, a version control technique able to properly manipulate uncertain data may be very helpful in this kind of applications. Unfortunately, current version control approaches do not allow to handle uncertain data. To try to resolve this problem, authors propose in [6] a probabilistic approach for version control able to efficiently handle uncertain data in the case of large-scale Web editing collaborative platforms.

In this model, uncertain data are handled through a probabilistic XML model [29], basic component of the proposed version control framework. Each version of a shared document is represented by an XML tree. At the inner level, each version of a document is associated with a random event that plays a similar role than the Id of a DBV in the DBV approach of [10]. An edit script is attached to each event and a directed acyclic graph of random events keeps track of the derivation history (succession of the events). Thus, a multiversion uncertain document is modeled as a probabilistic XML document, represented by an XML tree whose edges are annotated by propositional formulas over random events. Each propositional formula models both the semantics of uncertain editions (insertion and deletion) performed over a given part of the document and its provenance in the version control process.

Uncertainty is evaluated using the probabilistic model and the reliability measure associated with each data source, each contributor, or each editing event, resulting in an uncertainty measure on each version and each part of the document. The work shows that standard version control operations, especially the update operation, can be implemented directly as an operation on the probabilistic XML document. The efficiency of the proposed approach, with respect to deterministic version control systems like Git and Subversion, is demonstrated on real-world datasets in [6].

[2]http://www.wikipedia.org/
[3]https://drive.google.com/

3.6 Optimization of multiversion data locally

The efficient management of temporal and multiversion data is crucial for many traditional and emerging database applications. A major performance bottleneck for database systems is the memory hierarchy. One of the main means for optimizing the utilization of the memory hierarchy is to optimize data spatial locality, *i.e.* to put contiguously data that are likely to be read simultaneously. The problem studied in [22] is to optimize temporal and multiversion data spatial locality at all levels of the memory hierarchy, using index structures and storage policies. In particular, the authors in [23, 24, 26, 27] propose a cost model, the steady state analysis, allowing an accurate estimation of the performance of different index structures. The analysis provides database designers with tools allowing them to determine the most suitable index structure, for given data and application characteristics. The authors in [25, 28] also studies the impact of version redundancy on L2 cache utilization and propose new storage models which, in contrast with the standard storage models, avoid version redundancy and optimize L2 cache and main memory bandwidth utilization.

4 CONCLUSIONS AND NEW RESEARCH DIRECTIONS

Database versioning is still a current problem research. Recently, several version manager systems have been proposed: Decibel [34] and OrpheusDB [19] for versionning relational databases, AUDIT [40] relying on NoSQL BigData stores. Recent papers have also focused on index maintenance [46] or on storage and recreation costs [8] in multiversion databases.

As discussed in this paper, the DBV approach offers powerful mechanisms for managing multiversion databases. It can be applied to situations in which researchers need to manage the evolution of the world - e.g., smart cities, climate change, human migration, health. Proposed in 1990, it has recently been extended [5, 48] and in particular has been implemeneted into the MyDraft platform, developed by Karmicsoft, associated with the LAMSADE research laboratory at the Université Paris-Dauphine (France).

The power of the DBV model lies in its simplicity and genericity. As a consequence, there are still many open fronts that can profit from its use. One of them is provenance of computational experiments — those that can be expressed via scientific workflows. Scientific research is based on successive trial-and-error attempts, and thus appropriate modeling of workflow versions. On the theoretical side, there is still much to be done in maintaining consistency across multiple versions of the world, such as the new kinds of consistency constraints that are being studied in the context of graph databases and network science.

5 ACKNOWLEDGEMENTS

We would like to thank the many contributors to the works presented in this article. In particular, Mouhamadou Lamine Ba, Raymond Bisdorff, Luis Camargo, Gilles Dodinet, Anne Doucet, Marc Joliveau, Patrick Kabore, João Sávio C. Longo, Sophie Monties, Vincent Oria, Mohamed Ally Peerbocus, André Santanché, Pierre Senellart, Agnès Voisard, Florian de Vuyst, ...

Last but not least, we pay our tribute here to our visionnary, inspiring and kind colleague, advisor and dear friend, Genevieve Jomier.

REFERENCES

[1] Talel Abdessalem. 1997. *Approche des versions de bases de données : représentation et interrogation des versions.* Ph.D. Dissertation. Université Paris-Dauphine (France).

[2] Talel Abdessalem and Geneviève Jomier. 1997. VQL: A Query Language for Multiversion Databases. In *Int. Workshop on Database Programming Languages (DBPL), August 18-20, Estes Park, Colorado (USA).* 160–179.

[3] Talel Abdessalem, Claudia Bauzer Medeiros, Wojciech Cellary, Maude Manouvrier, Marta Rukoz, and Michel Zam. 2018. Les Versions de Bases de Données. *Livre Recherche des 50 ans de Dauphine* (2018). To appear.

[4] Kerstin Altmanninger, Martina Seidl, and Manuel Wimmer. A survey on model versioning approaches. *nt. J. of Web Information Systems* 5 (????), 271–304. Issue 3.

[5] Mouhamadou Lamine Ba, Talel Abdessalem, and Pierre Senellart. 2013. Merging Uncertain Multi-Version XML Documents. In *Int. Workshop on Document Changes: Modeling, Detection, Storage and Visualization, Florence, Italy, September 10.*

[6] M. Lamine Ba, Talel Abdessalem, and Pierre Senellart. 2013. Uncertain Version Control in Open Collaborative Editing of Tree-Structured Documents. In *ACM Symp. on Document Eng. (DocEng), Sept. 10 - 13, Florence (Italy).* 10–13.

[7] F Bancilhon, C Delobel, and P Kanellakis. 1992. Building an OODBMS, the story of O2. (1992).

[8] Anant Bhardwaj, Amol Deshpande, Aaron J Elmore, David Karger, Sam Madden, Aditya Parameswaran, Harihar Subramanyam, Eugene Wu, and Rebecca Zhang. 2015. Collaborative data analytics with DataHub. *VLDB Endowment (PVLDB)* 8, 12 (2015), 1916–1919.

[9] Raymond Bisdorff and Michel Zam. 2011. Modern MCDA software: requirements and opportunities. *Newsletter of the EURO Working Group Multicriteria Aid for Decisions* 3, 23 (2011), 1–2.

[10] Wojciech Cellary and Geneviève Jomier. 1990. Consistency of Versions in Object-Oriented Databases. In *Int. Conf. on Very Large Data Bases (VLDB), August 13-16, 1990, Brisbane (Australia).* 432–441.

[11] Scott Chacon. 2014. *Git Book (http://book.git-scm.com/).*

[12] Ben Collins-Sussman, Brian W. Fitzpatrick, and C. Michael Pilato. 2008. *Version Control with Subversion.* O'Reilly Media.

[13] Luis del Val Cura. 1997. *Processing versions in geographic databases (in Portuguese).* Master's thesis. Institute of Computing, University of Campinas – Unicamp (Brazil).

[14] Anne Doucet and Sophie Monties. 1997. Versions of integrity constraints in multiversion databases. In *Int. Conf. on Database and Expert Systems Applications (DEXA), Sept. 1-5, Toulouse (France).* Springer, 252–261.

[15] Stéphane Gançarski. 1999. Database Versions to Represent Bitemporal Databases. In *Int. Conf. on Database and Expert Systems Applications (DEXA), August 30 - Sept. 3, Florence, (Italy).* 832–841.

[16] Stéphane Gançarski and Geneviève Jomier. 1994. Managing Entity Versions within their Contexts: A Formal Approach. In *Int. Conf. on Database and Expert Systems Applications (DEXA), Sept. 7-9, Athens (Greece).* 400–409.

[17] Stéphane Gançarski and Geneviève Jomier. 2001. A framework for programming multiversion databases. *Data Knowl. Eng.* 36, 1 (2001), 29–53.

[18] Stéphane Gançarski. 1994. *Versions et bases de données; modèle formel, supports de langage et d'interface utilisateur.* Ph.D. Dissertation. Université Paris XI Orsay (France).

[19] Silu Huang, Liqi Xu, Jialin Liu, Aaron J. Elmore, and Aditya G. Parameswaran. 2017. OrpheusDB: Bolt-on Versioning for Relational Databases. *VLDB Endowment (PVLDB)* 10, 10 (2017), 1130–1141.

[20] Geneviève Jomier, Maude Manouvrier, Vincent Oria, and Marta Rukoz. 2005. Multi-level index for global and partial content-based image retrieval. In *Int. Conf. on Data Engineering Workshops (ICDEW) 5-8 April, Tokyo (Japan).* 66–75.

[21] Geneviève Jomier, Maude Manouvrier, and Marta Rukoz. 2000. Storage and management of similar images. *J. of the Brazilian Computer Society* 6 (00 2000), 13 – 25.

[22] Khaled Jouini. 2008. *Optimisation de la localité spatiale des données temporelles et multiversions.* Ph.D. Dissertation. Université Paris-Dauphine (France).

[23] Khaled Jouini and Geneviève Jomier. 2006. Comparaison de méthodes d'indexation des versions. In *Journées Bases de Données Avancées, (BDA), Lille (France).*

[24] Khaled Jouini and Geneviève Jomier. 2007. Indexing multiversion databases. In *ACM Conf. on Information and Knowledge Management (CIKM), Nov. 6-10, Lisbon (Portugal).* 915–918.

[25] Khaled Jouini and Geneviève Jomier. 2008. Avoiding version redundancy for high performance reads in temporal databases. In *Workshop on Data Management on New Hardware, (DaMoN), June 13, Vancouver (Canada).* 41–46.

[26] Khaled Jouini and Geneviève Jomier. 2009. Design and Analysis of Index Structures in MultiVersion Data. In *New Trends in Data Warehousing and Data Analysis.* Annals of Information Systems, Vol. 3. Springer, 1–21.

[27] Khaled Jouini and Geneviève Jomier. 2010. Modèles de stockage orientés interrogation pour bases de données temporelles. *Ingénierie des Systèmes d'Information* 15, 1 (2010), 61–85.

[28] Khaled Jouini, Geneviève Jomier, and Patrick Kabore. 2008. Read-Optimized, Cache-Conscious, Page Layouts for Temporal Relational Data. In *Int. Conf. on Database and Expert Systems Applications (DEXA), Sept. 1-5, Turin (Italy).* 581–595.

[29] Benny Kimelfeld and Pierre Senellart. 2013. Probabilistic XML: Models and Complexity. In *Advances in Probabilistic Databases for Uncertain Information Management*, Zongmin Ma and Li Yan (Eds.). Springer-Verlag.

[30] Al Koc and Abdullah Uz Tansel. 2011. A Survey of Version Control Systems. In *Int. Conf. on Engineering and Meta-Eng. (ICEME).*

[31] Joao S. C. Longo. 2013. *Management of Integrity Constraints for Multi-scale geospatial Data.* Master's thesis. Institute of Computing, University of Campinas – Unicamp (Brazil). Supervisor C. B. Medeiros.

[32] Joao Savio C. Longo, Luis Camargo, Claudia Bauzer Medeiros, and Andre Santanche. 2012. Using the DBV model to maintain versions of multi-scale geospatial data. In *Int. Workshop on Semantic and Conceptual Issues in GIS (SeCoGIS), LNCS 7518, Oct. 15-18, Florence (Italy).*

[33] Joao Savio C. Longo and Claudia Bauzer Medeiros. 2013. Providing multi-scale consistency for multi-scale geospatial data . In *Int. Conf. on Scientific and Statistical Database Management (SSDBM), July 29 - 31, Baltimore, MD (USA).*

[34] Michael Maddox, David Goehring, Aaron J Elmore, Samuel Madden, Aditya Parameswaran, and Amol Deshpande. 2016. Decibel: The relational dataset branching system. *VLDB Endowment (PVLDB)* 9, 9 (2016), 624–635.

[35] Maude Manouvrier. 2000. *Objets Similaires de Grande Taille dans les Bases de Données.* Ph.D. Dissertation. Université Paris-Dauphine (France).

[36] Maude Manouvrier, Marta Rukoz, and Geneviève Jomier. 2002. Quadtree representations for storage and manipulation of clusters of images. *Image Vision Comput.* 20, 7 (2002), 513–527.

[37] Claudia Bauzer Medeiros, Marie-Jo Bellosta, and Genevieve Jomier. 1996. Managing Multiple Representations of Georeferenced Elements. In *Int. Workshop on Database and Expert Systems Applications (DEXA), Sept. 9-10, Zurich (Switzerland).* 364–371.

[38] Claudia Bauzer Medeiros, Marc Joliveau, Genevieve Jomier, and Florian de Vuyst. 2010. Managing sensor traffic data and forecasting unusual behaviour propagation. *Geoinformatica* 14 (2010), 279–305.

[39] Claudia Bauzer Medeiros and Genevieve Jomier. 1993. Managing Alternatives and Data Evolution in GIS. In *ACM Workshop on Advances in Geographic Information Sys.* 36–39.

[40] Khaleel Mershad, Qutaibah M Malluhi, Mourad Ouzzani, Mingjie Tang, Michael Gribskov, and Walid G Aref. 2018. AUDIT: approving and tracking updates with dependencies in collaborative databases. *Distributed and Parallel Databases* 36, 1 (2018), 81–119.

[41] Sophie Monties. 1997. *Cohérence des bases de données multi versions orientées objet.* Ph.D. Dissertation. Université Paris 1/Panthéon-Sorbonne (France).

[42] Mohammed-Ally Peerbocus. 2001. *Gestion de l'évolution spatiotemporelle dans une base de données géographiques.* Ph.D. Dissertation. Université Paris-Dauphine (France).

[43] Mohamed Ally Peerbocus, Claudia Bauzer Medeiros, Geneviève Jomier, and Agnès Voisard. 2004. A System for Change Documentation Based on a Spatiotemporal Database. *GeoInformatica* 8, 2 (2004), 173–204.

[44] Mateus S. Pierre. 2007. *Access Control in Multiversion Databases (in Portuguese.* Master's thesis. Institute of Computing, University of Campinas – Unicamp (Brazil).

[45] Marta Rukoz, Maude Manouvrier, and Geneviève Jomier. 2006. Delta-distance: A family of dissimilarity metrics between images represented by multi-level feature vectors. *Inf. Retr.* 9, 6 (2006), 633–655.

[46] Mohammad Sadoghi, Kenneth A Ross, Mustafa Canim, and Bishwaranjan Bhattacharjee. 2016. Exploiting SSDs in operational multiversion databases. *The VLDB Journal* 25, 5 (2016), 651–672.

[47] Hanan Samet. 1984. The quadtree and related hierarchical data structures. *ACM Computing Surveys (CSUR)* 16, 2 (1984), 187–260.

[48] André Santanchè, João Sávio C. Longo, Geneviève Jomier, Michel Zam, and Claudia Bauzer Medeiros. 2014. Multi-focus Research and Geospatial Data - anthropocentric concerns. *J. of Information and Data Management (JIDM)* 5, 2 (2014), 146–160.

[49] Agnès Voisard, Claudia Bauzer Medeiros, and Genevieve Jomier. 2000. Database Support for Cooperative Work Documentation. In *Int. Conf. on the Design of Cooperative Systems (COOP), May 23-26, Sophia-Antipolis (France)*. 275–290.
[50] Michel Zam. 1998. *Contribution à la traçabilité du processus de conception en génie logiciel*. Ph.D. Dissertation. Université Paris IX Dauphine (France).
[51] Michel Zam. 2015. User Experience Driven Design of MCDA Problems with DecisionCloud. *Newsletter of the EURO Working Group Multicriteria Aid for Decisions* 3, 32 (2015), 7–10.
[52] Michel Zam, Gilles Dodinet, and Geneviève Jomier. 2011. Software objects fairy tales: merging design and runtime objects into the cloud with mydraft. In *Annual ACM SIGPLAN Conf1; on Object-Oriented Programming, Systems, Languages, and Applications, (OOPSLA), Oct. 22 - 27, Portland, OR (USA)*. 43–44.