A Cross-Domain Survey on Coupled Evolution
– List of Excluded Papers –

Markus Herrmannsdoerfer¹, Sander D. Vermolen², Guido Wachsmuth²

1 Institut für Informatik, Technische Universität München
Boltzmannstr. 3, 85748 Garching b. München, Germany
herrmann@in.tum.de
2 Software Engineering Research Group, Delft University of Technology
The Netherlands
{S.D.Vermolen, G.H.Wachsmuth}@tudelft.nl

1 Excluded Papers Within the Relevant Domain

Following is a list of papers, which were within the domain of the survey, yet were rejected as they fall outside the scope definition. For each publication, we list its bibliographic information, followed by the reason for rejection.

   About semantic adaptation of schema mappings
   Hardly any coverage of database migration
   About avoiding migration rather than supporting migration
   Not about migration of instances
   Instance of coupled evolution of grammars and transformations
   Main focus of the paper not about coupled evolution, coupled evolution only a very small part

About revalidation after schema evolution


About information integration


Does not cover evolution


About integrity checking rather than migration


On API evolution


On application evolution (limited conformance relation)


On API Evolution


No instance migration, but schema migration, in response to instance change


About preventing coupled evolution instead of supporting coupled evolution


About preventing coupled evolution instead of supporting coupled evolution

*About serialization in OO programming*

Philip A. Bernstein Applying Model Management to Classical Meta Data Problems

*No migration of elements*


Similar to the included:


Several extensions for ORION, one is schema evolution, but only without real migration


Several extensions for ORION, one is schema evolution, but only without real migration


Summary of a panel discussion, out of scope


Only a short tool description


Preserve well-formedness during schema evolution, no co-evolution

Evolution of objects (and their classification) in the extension, no evolution of the classification scheme


Propagation of evolution from conceptual to logical schemas to extensions and programs, no co-evolution


On data versioning, not database or schema versioning


On schema evolution, which does not affect existing data, thus preventing the need for database migration, thus not on coupled evolution

- E. Bertino A view Mechanism for Object-Oriented Databases in EDBT 136-151 1992

On views, no focus on coupling


Focused on preventing coupled evolution


On data versioning, not meta data versioning


Primarily focused on versions and change notifications, only briefly mentions dynamic schema evolution, no coupling


Only mentions class evolution as problem in software communities, no real approach


A general description of ORION, it only briefly discusses dynamic schema evolution in ORION. Not relevant enough for the survey, the ORION features are already discussed in other included papers

*Does not focus on coupled evolution, only mentions it as an issue in model management*

– (in CAiSE 2008) Vasilios Andrikopoulos, Salima Benbernou, Mike P. Papazoglou: Managing the Evolution of Service Specifications. 359-374

*Extension is not completely defined by the intensional definition*


*Does not address coupled evolution, rather system evolution in general*

– (in ER 1996) Moira C. Norrie, Andreas Steiner, Alain Wunderli A Model for Classification Structures with Evolution Control. 456-471

*Focus on evolution of objects, not their structure description. Thereby no vertical coupling.*


*Focus on evolution of data, not their schema. Thereby no vertical coupling.*


*Focus on entity evolution, no instance adaptation: no coupling*


*Focus on schema evolution, no database migration: no coupling*


*About evolution of schema integration framework on evolution of schema modeling language, such that schema integration can be reused: Extension not completely defined by intensional definition*


*Evolution of object-oriented software, about recommending refactorings*


*Migration of runtime C++ objects from a node to another in a distributed setting*


*Migration of objects in a running application when their classes change*

Integration of legacy systems into modern systems


Integration of legacy systems into modern systems


Describes a library of transformations on class diagrams, but without the co-transformation of instances


Avoids migration of programs written with the evolving DSL


Migration of legacy data into modern database systems


Introduces evolutionary language design, but does not provide an approach to migrate language utterances


Support the addition of context-sensitive rules by dynamically evolving the grammar when parsing a program


Migration of legacy data to modern database systems


Effort for program evolution rather than that for language evolution


About load balancing during migration rather than about migration itself

- Li, Q., McLeod, D.: Conceptual database evolution through learning in object databases. IEEE Trans. on Knowl. and Data Eng. 6(2) (1994) 205–224

The paper discusses the direction of ontoloware and does not explicitly talk about instance migration

Migration of queries in response to schema evolution


Migration of queries in response to schema evolution


Markowitz et al. state: “We assume in this paper that the database state is empty. The coupling of schema restructuring manipulations with state mappings is investigated in [43].”


Data migration is not discussed


Hainaut states in the conclusion: “Several problems still are to be addressed ... How can data structure transformations be propagated to the other components of the information system, notably the data (data conversion)?”


Peters et al. state: “Due to space restrictions, change propagation is not addressed in this paper.”


On preserving schema consistency during evolution


Paper focuses on evolution and versioning of elements (objects) rather than the intensional definition (schema)

2 Excluded Papers Outside the Relevant Domain

This section enumerates the papers we rejected for falling outside the considered domain. Each subsection is devoted to a particular domain. Note that these are not intended to be complete lists for any of the domains, they merely enumerate the papers considered for the survey.

2.1 Process Evolution

2.2 Difference calculation & representation


2.3 Schema Matching & Integration


2.4 Software Evolution

2.5 Ontology Evolution