Preface to the Third International Workshop on Bidirectional Transformations

Soichiro Hidaka
National Institute of Informatics, Japan
hidaka@nii.ac.jp

James F. Terwilliger
Microsoft Research
james.terwilliger@microsoft.com

ABSTRACT

This workshop is the third in a series promoting cross-collaboration between computer science disciplines on the topic of bidirectional transformations. The workshop had a 53% acceptance rate from submissions from four different disciplines. In this brief preface, we outline a definition for what makes a bidirectional transformation, and describe a history of the workshop and its associated meeting series.

Keywords

Bidirectional Transformations, Cross-Disciplinary

1. INTRODUCTION

The workshop series on Bidirectional Transformations (BX) is dedicated to bringing researchers together from different disciplines of computer science working on BX-related projects. There are at least four disciplines actively or historically working on such projects:

1. *Databases*, whose history of work on updateable views [2, 4] often serves as the semantic backbone for other work both within and without the field, and whose work on data exchange represents an opportunity to apply such theory to new applications [1].
2. *Programming languages*, whose work on lenses [5] often form the formal basis for new work.
3. *Graph transformations*, whose work on triple graph grammars [8] serves as a way to grow transformations from simple rules.
4. *Software engineering*, whose work stems from a need to manage significant numbers of software artifacts in practical settings.

We accepted 9 out of 17 submissions. Each of the four sub-disciplines was represented by at least one submitted paper.

Previous workshops were held with the European Joint Conferences on Theory and Practice of Software (ETAPS), a federation of smaller conferences tailored to several disciplines of computer science but not a database audience. As the intent of the workshop series is to rotate through venues to promote communication across disciplines, the steering committee made the decision to hold this year's instance in association with a database conference for the first time.

The next BX workshop will likely be held with the Software Technologies: Applications and Foundations (STAF) federation of conferences. Among the many conferences associated with STAF is the International Conference on Model Transformations (ICMT). ICMT had often been a frequent venue for papers on BX, especially prior to the start of the workshop series, so associating with STAF next year will a good fit for the workshop and a way for the BX workshop to return home for a year, after a fashion.

2. WHAT IS BX?

Bidirectional transformations are a mechanism to maintain consistency between two (or more) related sources of information [3]. One can think of a bidirectional transformation as a pair of transformations in opposite directions. Suppose one is given two sets S and T of artifacts such as strings, trees, or tables. For a given source artifact $s \in S$, the *forward transformation* $f : S \to T$ produces a target artifact $t \in T$, while a *backward transformation* $g : T \to S$ produces a source from a target. To achieve consistency using the bidirectional transformations, f and g are required to satisfy certain round-tripping properties.

The obvious case is when $g = f^{-1}$. However, this case is often considered to be too restrictive. In particular, f is not allowed to lose information. To overcome this restriction, g is allowed to access the original source artifact, making the function binary, i.e., $g : S \times T \to S$. This round-tripping property, or *well-behavedness*, can be characterized as follows, giving names *get* and *put* to the forward and backward transformations [5]:

$$\forall s \in S.\ \mathit{put}\,(\mathit{get}\,s, s) = s \qquad \text{(GetPut)}$$

$$\forall s \in S, \forall t \in T.\ \mathit{get}\,(\mathit{put}\,(t, s)) = t \qquad \text{(PutGet)}$$

GetPut says that you can come back to the original source if there is no change on the target, while PutGet says that all information in the target are propagated to the source (therefore the *get* can always recover the updated target).

It is worth noting that the above formulation corresponds to the properties for the view update problem [2]. The view update problem is, given a database state $s \in S$ and a query $f : S \to T$ and an update $u : T \to T$ on the view $f(s)$, translating u to the update $T_u : S \to S$ on the database.

Then one can formulate the desirable properties as:

$$\forall s \in S.\ uf(s) = f(s) \Rightarrow T_u(s) = s \qquad \text{(Acceptability)}$$
$$uf = fT_u. \qquad \text{(Consistency)}$$

Acceptability says if there is no update on the view, then there will be no update on the database, corresponding to GetPut. On the other hand, Consistency says the updated view $uf(s)$ can be reconstructed by regenerating a view from the database on which the translated update T_u has been applied. Therefore it corresponds to PutGet.

The scheme above is asymmetric in that *put* was binary while *get* remained unary. In graph transformation and its application to software engineering, we often see a symmetric scheme where the old target is used in the forward transformation. Consistency is no longer represented by a function, but by a relation $R \subset S \times T$. $(s, t) \in R$ if and only if $s \in S$ and $t \in T$ are consistent. If the forward and backward transformation are denoted by $\overrightarrow{T}$ and $\overleftarrow{T}$ respectively, then well-behavedness is expressed by:

$$(s,t) \in R \Rightarrow \overrightarrow{T}(s,t) = t \wedge \overleftarrow{T}(s,t) = s \qquad \text{(Hippocraticness)}$$
$$(s, \overrightarrow{T}(s,t)) \in R \wedge (\overleftarrow{T}(s,t), t) \in R. \qquad \text{(Correctness)}$$

Hippocraticness says that if the forward transformation is applied to an already consistent pair of source and target, the same artifact (original target for $\overrightarrow{T}$, or original source for $\overleftarrow{T}$) is obtained, so it corresponds to GetPut in the asymmetric setting. On the other hand, Correctness says that the result of the transformations are always consistent. It corresponds to PutGet in the sense that for updated target t, source s that satisfies *get* $s = t$ is always obtained.

Different approaches and implementations often refer to different notions of correctness properties, and community-wide efforts have been made to share the notions with examples, some of which also appear in this volume.

3. HISTORY AND CONTEXT

Since 2008, researchers from the four disciplines referenced in Section 1 have been meeting periodically with the intention of opening up communication between those fields and potentially establishing a common research agenda.

Shonan: December, 2008

The first meeting was in Shonan near Tokyo in 2008 [3] and served primarily as an introduction. At that time, most of the participants had little-to-no exposure to the research going on in other disciplines. Most of the meeting was spent with participants introducing their own work, or relevant research with which they are familiar. By the end of the week, the participants collectively decided that there was significant overlap between that work, enough to merit further discussion. They decided that such work should be citing each other more, and there could be some interesting collaboration and unification to be done. Work began to arrange another meeting to follow up on possible collaborations.

Dagstuhl: January, 2011

A second meeting occurred in early 2011 at Dagstuhl [6]. The meeting began with representatives from each discipline giving short-form tutorials of 2–3 hours on the tools from that discipline to bring people up to speed quickly. The tutorials presented not only the bidirectional problems intrinsic to that discipline and the primary tools in its solution space, but also that discipline's requirements for formal properties of BX. There was ample time for new participants to present their own work. Finally, there was space in the schedule left open for group work and discussion.

The primary work product of the Dagstuhl meeting was the establishment of the BX workshop series itself, whose first instance was the following year in Tallinn, Estonia.

Banff: December, 2013

Most recently, another meeting was held in late 2013 in Banff, Canada (summary publication forthcoming). The meeting contained some short discipline-specific tutorials again, but most of the time was dedicated to working group and breakout sessions. These sessions covered a number of topics, but two of the most populated and productive focused on how to benchmark BX tools and how to put together a repository of examples. Both of those discussions yielded work that is published in this workshop proceedings.

What has become clear over the course of the meetings is that it is difficult in any collaboration to get past the point where people primarily talk about their own work. Defining a metric of success for this sequence of meetings — and this workshop series as well, for that matter — is difficult, but almost certainly must include an increase in cross-disciplinary citation rates. It is not necessarily the case that we will eventually arrive at a central, unified research agenda, but at the very least we hope to see far more opportunities for collaborative research and publications (e.g., [7, 9]).

Another meeting is being planned at the time of this publication. It will likely happen sometime in 2015 or 2016. Most of the details are still in development.

4. REFERENCES

[1] M. Arenas, P. Barceló, L. Libkin, and F. Murlak. Relational and XML Data Exchange. Morgan and Claypool Publishers, 2010.

[2] Bancilhon, F. and Spyratos, N.: Update Semantics of Relational Views, *ACM Transactions on Database Systems*, December 1981, 6(4).

[3] K. Czarnecki, J. N. Foster, Z. Hu, R. Lämmel, A. Schürr, and J. F. Terwilliger. Bidirectional Transformations: A Cross-Discipline Perspective. *ICMT 2009*.

[4] U. Dayal and P. Bernstein. On the Correct Translation of Update Operations on Relational Views. *ACM Transactions on Database Systems*, September 1982, 8(3).

[5] J. N. Foster, M. B. Greenwald, J. T. Moore, B. C. Pierce, and A. Schmitt. Combinators for bi-directional tree transformations: a linguistic approach to the view update problem. *POPL 2005*, 233–246.

[6] Z. Hu, A. Schürr, P. Stevens, and J. F. Terwilliger. Dagstuhl seminar on bidirectional transformations (BX). *SIGMOD Record* 40(1), (2011).

[7] M. Johnson, J. Pérez, and J. F. Terwilliger. What Can Programming Languages Say About Data Exchange? *EDBT 2014*.

[8] A. Schürr. Specification of graph translators with triple graph grammars. *WG '94* , pages 151–163. June 1995.

[9] J. F. Terwilliger, A. Cleve, and C. Curino. How Clean Is Your Sandbox? *ICMT 2012*.