A Survey of Stream Processing Languages

Martin Hirzel
IBM Research AI, USA
hirzel@us.ibm.com

Guillaume Baudart
IBM Research AI, USA
Guillaume.Baudart@ibm.com

Angela Bonifati
Lyon 1 University, France
angela.bonifati@univ-lyon1.fr

Emanuele Della Valle
Politecnico di Milano, Italy
emanuele.dellavalle@polimi.it

Sherif Sakr
KSAU-HS and UNSW
ssakratcse.unsw.edu.au

Akrivi Vlachou
University of Piraeus, Greece
avlachou@aueb.gr


```
1 SELECT IStream(Max(len) AS mxl,
2                MaxCount(len) AS num,
3                ArgMax(len, caller ) as who)
4 FROM Calls[Range 24 Hours Slide 1 Minute]
```

Figure 1: CQL code example.

ABSTRACT

TODO

1. INTRODUCTION

```
TODO ~0.8 pages
- motivation
  - importance and rise of streaming
  - benefits of languages
  - diversity of languages, lack of standard
  - lack of a recent survey [16] [14]
- background / definitions
  - stream, streaming application, DSL, stream processing language
  - EDSLs [13], recent trends in EDSLs
  - requirements: performance, generality, productivity
  - most streaming languages are declarative, so the traditional
    paradigm categories imperative or functional don't apply
- roadmap for rest of paper
- take-home message: democratization of streaming
```

2. STREAM PROCESSING LANGUAGES

```
TODO ~2.8 pages total
- this section will have ~0.3 pages about each language, and
  each of these snippets will be structured around questions:
  why-who-when-what-where-whence
  (see CQL for an example, but the others will be similar!)
- before the snippets on each of the languages, we will briefly
  introduce and explain these questions, as follows:
  - why: objective, audience, domain
  - who: inventors, supporters
  - when: first release / first paper
  - what: key idea, data model, type system, code example
  - where: being developed or offered today, license
  - whence: influenced-by and influences
- descriptions of individual languages
  - relational / CQL [2]   Sherif
    - why: precise semantics [4], example app linear-road [3]
    - who: Arasu/Widom at Stanford
    - when: 2004
    - what: algebra of R2R (relational), S2R (windows), R2S (I/R/D);
      data model relational; type system [15]; example Figure~1
    - where: Stanford STREAM no longer active
    - whence: influenced-by TelegraphCQ [8],
      influences StreamInsight [1] and many others
  - synchronous dataflow / Lustre [7] Guillaume
  - big-data streaming / SPL [12] Martin, Emanuele
  - CEP (complex event processing) / MATCH-RECOGNIZE [18] [11]
  - XML / NiagaraCQ [9], YFilter [10] Sherif
  - RDF / C-SPARQL [6] Emanuele, Akrivi
  - stream reasoning Emanuele
  - reactive spreadsheets / ActiveSheets [17] Martin
  - controlled natural language / META [5] Martin
- close with summary/comparison table
  - perhaps according to performance/generality/productivity
```

3. WHAT'S NEXT?

```
TODO ~0.6 pages total
- this section will have ~0.2 pages about each challenge,
  where the challenges come from our Dagstuhl discussion,
  and each snippet will be structured as follows:
  - what's the problem
  - why is solving it useful
  - why is it hard
  - what makes it a streaming languages problem
- descriptions of individual challenges:
  - Handling data variety while keeping the language
    simple and fast Emanuele
  - Handling veracity in a simple and well-defined way
    Akrivi
  - Getting broad adoption for a streaming language
```

4. CONCLUSION

```
TODO ~0.2 pages
```


Acknowledgements

The idea for this paper was conceived at Dagstuhl Seminar 17441 on "Big Stream Processing Systems".

5. REFERENCES

[1] M. H. Ali, C. Gerea, B. Raman, B. Sezgin, T. Tarnavski, T. Verona, P. Wang, P. Zabback, A. Kirilov, A. Ananthanarayan, M. Lu, A. Raizman, R. Krishnan, R. Schindlauer, T. Grabs, S. Bjeletich, B. Chandramouli, J. Goldstein, S. Bhat, Y. Li, V. Di Nicola, X. Wang, D. Maier, I. Santos, O. Nano, and S. Grell. Microsoft CEP server and online behavioral targeting. In *Demo at Very Large Data Bases (VLDB-Demo)*, pages 1558–1561, 2009.

[2] A. Arasu, S. Babu, and J. Widom. The CQL continuous query language: semantic foundations and query execution. *Journal on Very Large Data Bases (VLDB J.)*, 15(2):121–142, 2006.

[3] A. Arasu, M. Cherniack, E. Galvez, D. Maier, A. S. Maskey, E. Ryvkina, M. Stonebraker, and R. Tibbetts. Linear road: A stream data management benchmark. In *Conference on Very Large Data Bases (VLDB)*, pages 480–491, 2004.

[4] A. Arasu and J. Widom. A denotational semantics for continuous queries over streams and relations. *SIGMOD Record*, 33(3), Sept. 2004.

[5] M. Arnold, D. Grove, B. Herta, M. Hind, M. Hirzel, A. Iyengar, L. Mandel, V. Saraswat, A. Shinnar, J. Siméon, M. Takeuchi, O. Tardieu, and W. Zhang. META: Middleware for events, transactions, and analytics. *IBM Journal of Research and Development*, 60(2–3):15:1–15:10, 2016.

[6] D. F. Barbieri, D. Braga, S. Ceri, E. Della Valle, and M. Grossniklaus. C-SPARQL: SPARQL for continuous querying. In *Poster at International World Wide Web Conferences (WWW-Poster)*, pages 1061–1062, 2009.

[7] P. Caspi, D. Pilaud, N. Halbwachs, and P. Raymond. LUSTRE: a declarative language for real-time programming. In *Symposium on Principles of Programming Languages (POPL)*, pages 178–188, 1987.

[8] S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein, W. Hong, S. Krishnamurthy, S. Madden, V. Raman, F. Reiss, and M. A. Shah. TelegraphCQ: continuous dataflow processing for an uncertain world. In *Conference on Innovative Data Systems Research (CIDR)*, 2003.

[9] J. Chen, D. J. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continuous query system for internet databases. In *International Conference on Management of Data (SIGMOD)*, pages 379–390, 2000.

[10] Y. Diao, P. M. Fischer, M. J. Franklin, and R. To. YFilter: Efficient and scalable filtering of XML documents. In *Demo at International Conference on Data Engineering (ICDE-Demo)*, pages 341–342, 2002.

[11] M. Hirzel. Partition and compose: Parallel complex event processing. In *Conference on Distributed Event-Based Systems (DEBS)*, pages 191–200, 2012.

[12] M. Hirzel, S. Schneider, and B. Gedik. SPL: An extensible language for distributed stream processing. *Transactions on Programming Languages and Systems (TOPLAS)*, 39(1):5:1–5:39, March 2017.

[13] P. Hudak. Modular domain specific languages and tools. In *International Conference on Software Reuse (ICSR)*, pages 134–142, 1998.

[14] W. M. Johnston, J. R. P. Hanna, and R. J. Millar. Advances in dataflow programming languages. *ACM Computing Surveys (CSUR)*, 36(1):1–34, 2004.

[15] R. Soulé, M. Hirzel, B. Gedik, and R. Grimm. River: An intermediate language for stream processing. *Software – Practice and Experience (SP&E)*, 46(7):891–929, July 2016.

[16] R. Stephens. A survey of stream processing. *Acta Informatica*, 34(7):491–541, 1997.

[17] M. Vaziri, O. Tardieu, R. Rabbah, P. Suter, and M. Hirzel. Stream processing with a spreadsheet. In *European Conference on Object-Oriented Programming (ECOOP)*, pages 360–384, 2014.

[18] F. Zemke, A. Witkowski, M. Cherniak, and L. Colby. Pattern matching in sequences of rows. Technical report, ANSI Standard Proposal, 2007.