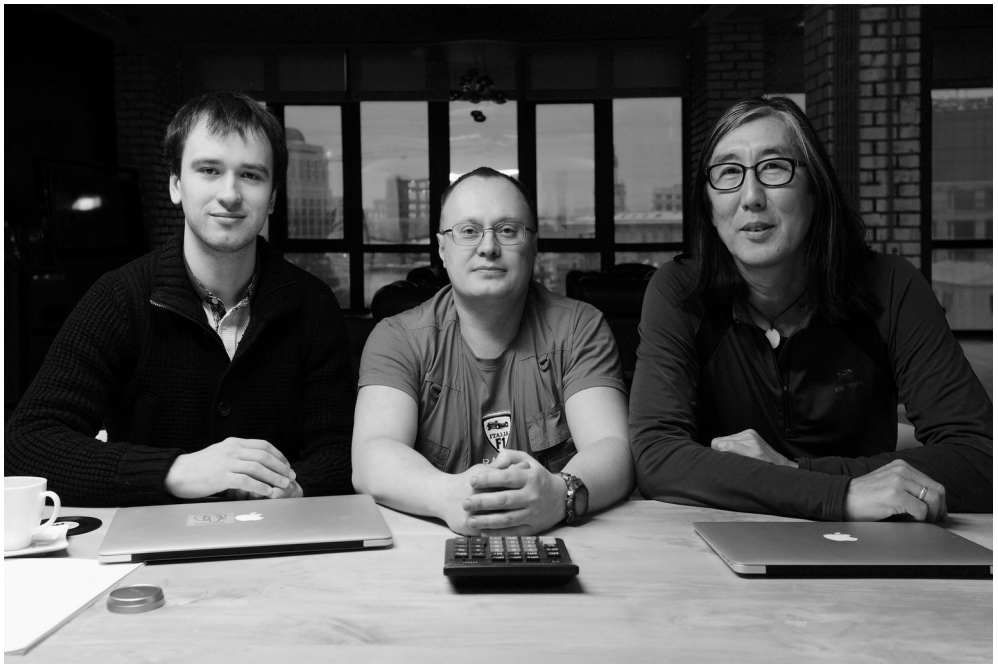
Сага о кластере

Олег Бартунов, Александр Коротков, Фёдор Сигаев
Postgres Professional

Highload++ 2015, Москва

Российские разработчики

Олег Бартунов, Федор Сигаев, Александр Коротков



PostgreSQL CORE

- Locale support
- PostgreSQL extendability:
- GiST(KNN), GIN, SP-GiST
- Full Text Search (FTS)
- NoSQL (hstore, jsonb)
- Indexed regexp search
- VODKA access method (WIP)

Расширения:

- Intarray
- Pg_trgm
- Ltree
- Hstore
- plantuner
- JsQuery

- Докладчики PGCon, PGConf: 20+ докладов
- Менторы GSoC
- Коммитеры PostgreSQL (1+1 in progress)
- Организаторы конференций
- 50+ лет экспертизы PostgreSQL: разработка, аудит, консалтинг
- Novartis, Raining Data, Heroku, Engine Yard, WarGaming, Rambler, Avito, 1c

Распределенные несчастия

2PC — не серебряная пуля

Распределенные несчастия: #1

Распределенные несчастия: #1

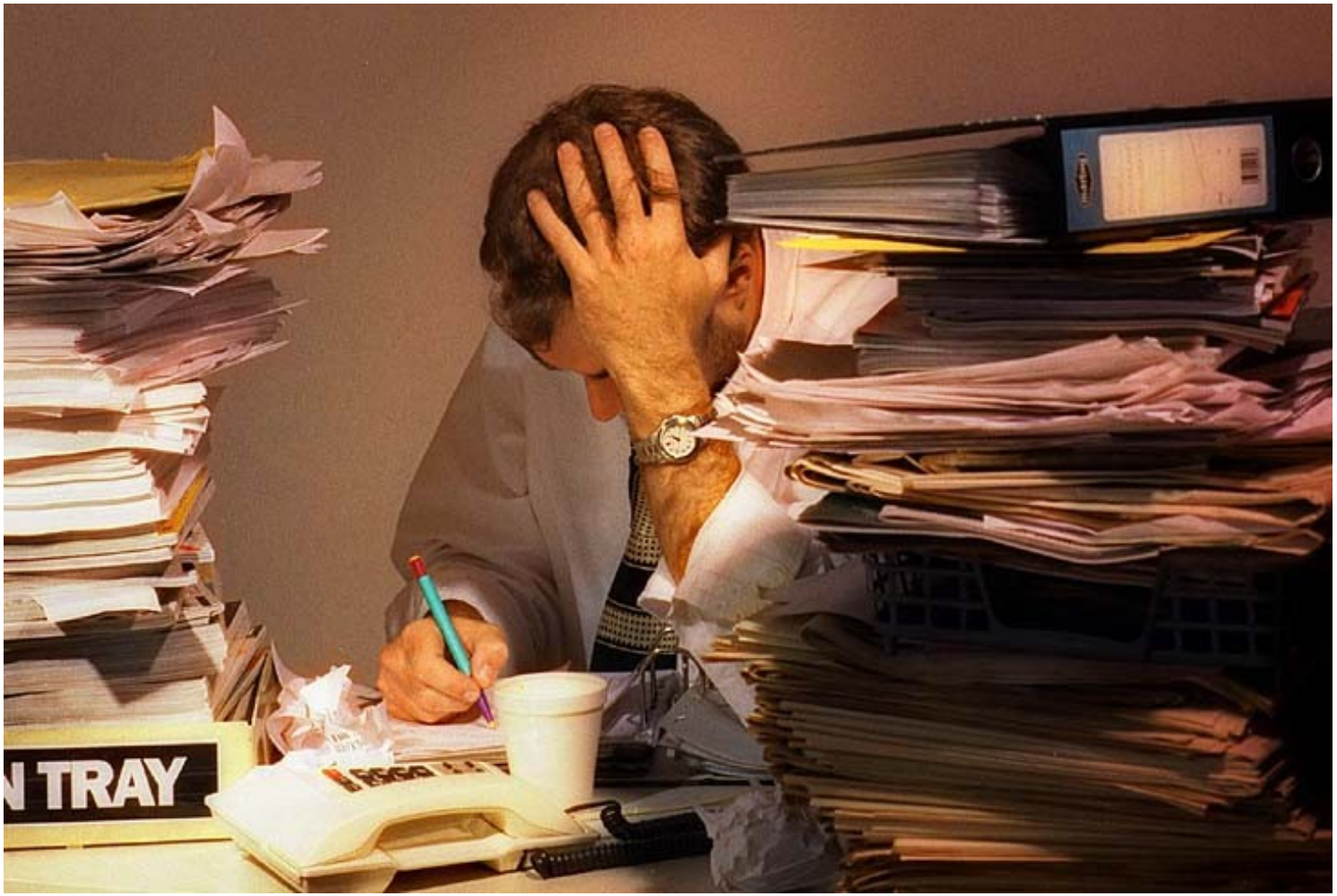
Решение:

Сделай сам или **добавь арбитра**

Распределенные несчастия: #2

Распределенные несчастия: #2

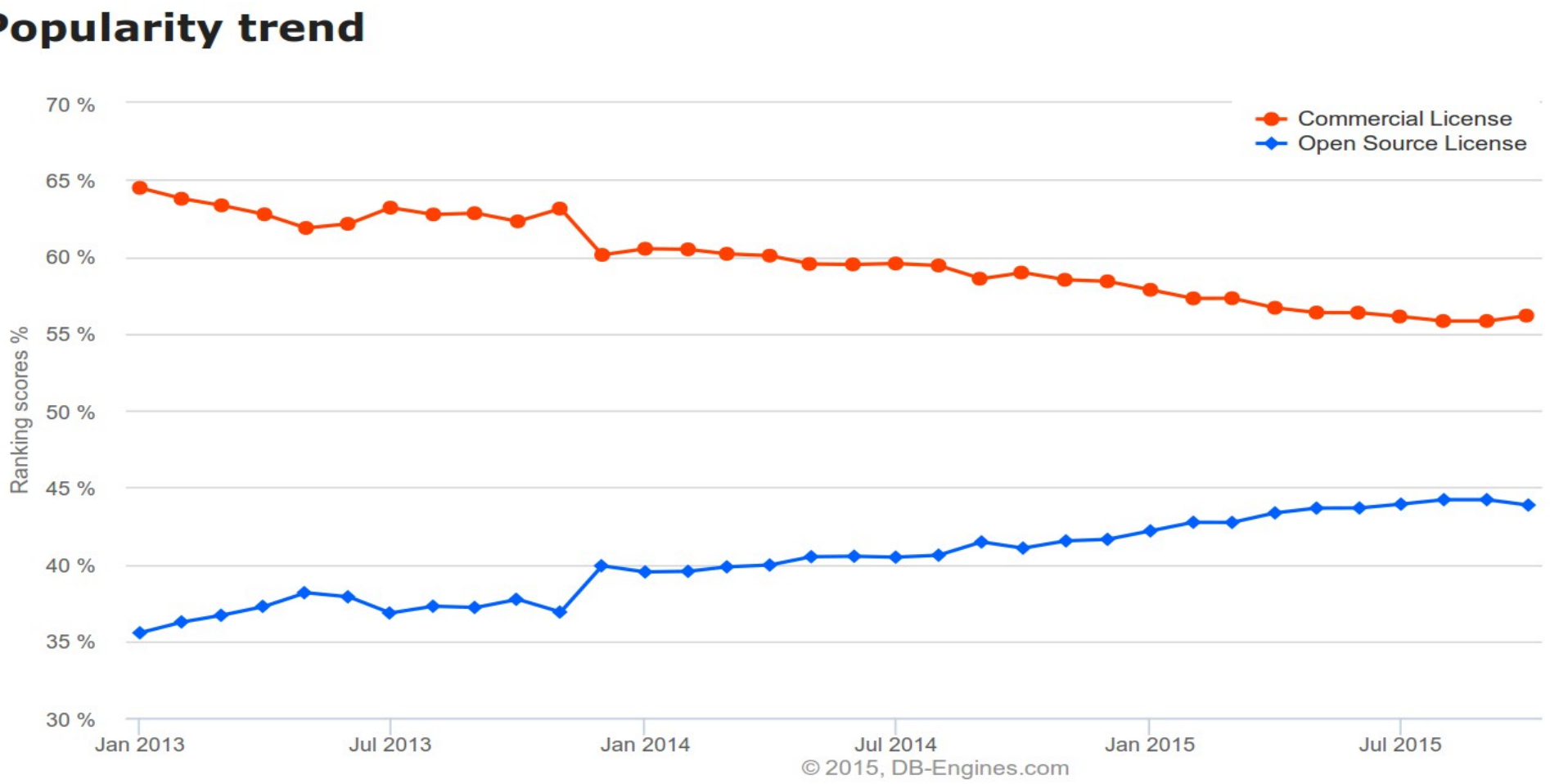
Вывод:
Целостный read не имеет решения в 2PC



Postgres PROFESSIONAL

Политические игры

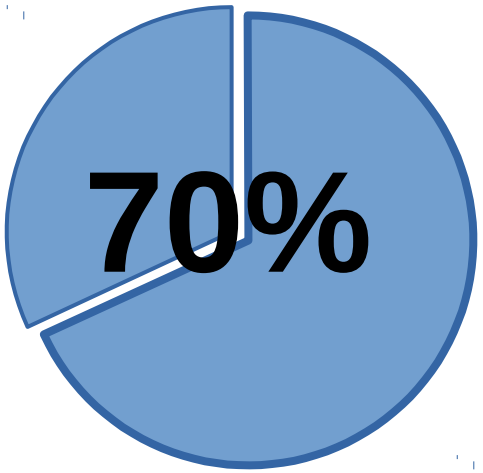
Мировые тенденции ПО



PROFESSIONAL Postgres

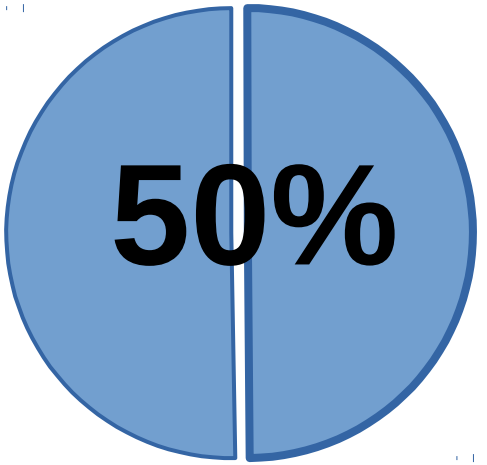
Gartner's Magic Quadrant

Gartner.Тенденции рынка СУБД



новых приложений

И



существующих приложений

будут использовать
реляционные базы данных
с **открытым кодом**
в 2018 году*

* Gartner, State of Open Source RDBMS, 2015, Donald Feinberg and Merv Adrian, April 21, 2015

Этапы развития PostgreSQL

Базовая функциональность
- JDBC
- MVCC
- Optimizer Stats
- PL/pgSQL

Стабилизация
- Исправление сбоев в работе
- Очистка кода
- Культура совершенства

Стандарты
- SQL 92 Joins
- Prepared queries
- Foreign Keys

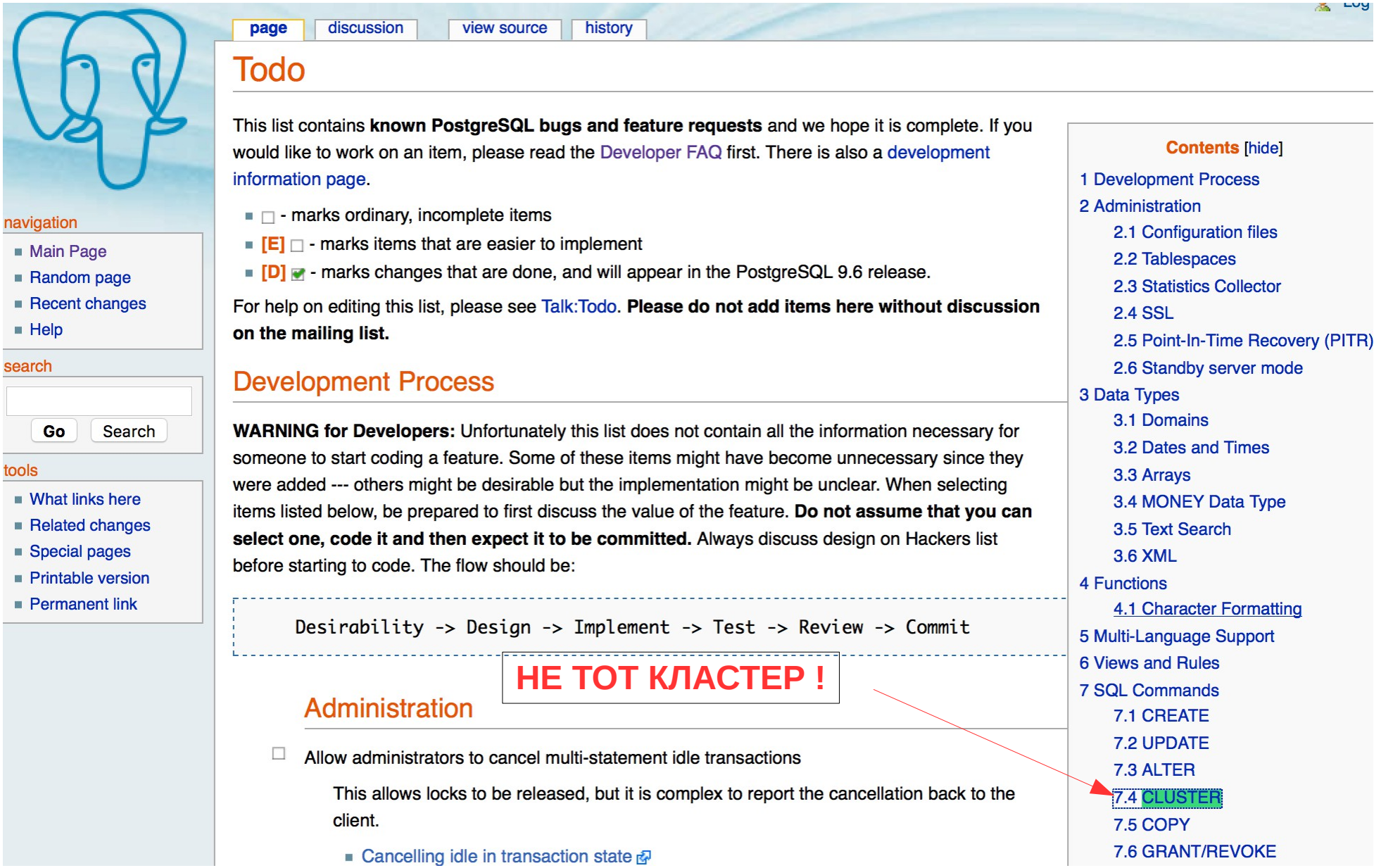
Функциональность ядра
- Write Ahead Log
- Prepared Queries
- Info. Schema
- Auto Vacuum

Возможности уровня Enterprise
- Потоковая репликация
- Производительность
- Вертикальное масштабирование
- PITR
- pg_upgrade
- NoSQL
- BDR
- Параллелизм

Простота использования
- Портирование на Windows
- pg_basebackup
- Различные инструменты

Community Todo: No Cluster !

https://wiki.postgresql.org/wiki/Todo



page | discussion | view source | history

Todo

This list contains **known PostgreSQL bugs and feature requests** and we hope it is complete. If you would like to work on an item, please read the Developer FAQ first. There is also a development information page.

- ☐ - marks ordinary, incomplete items
- **[E]** ☐ - marks items that are easier to implement
- **[D]** ☑ - marks changes that are done, and will appear in the PostgreSQL 9.6 release.

For help on editing this list, please see Talk:Todo. **Please do not add items here without discussion on the mailing list.**

navigation
- Main Page
- Random page
- Recent changes
- Help

search
Go Search

tools
- What links here
- Related changes
- Special pages
- Printable version
- Permanent link

Contents [hide]

1 Development Process
2 Administration
2.1 Configuration files
2.2 Tablespaces
2.3 Statistics Collector
2.4 SSL
2.5 Point-In-Time Recovery (PITR)
2.6 Standby server mode
3 Data Types
3.1 Domains
3.2 Dates and Times
3.3 Arrays
3.4 MONEY Data Type
3.5 Text Search
3.6 XML
4 Functions
4.1 Character Formatting
5 Multi-Language Support
6 Views and Rules
7 SQL Commands
7.1 CREATE
7.2 UPDATE
7.3 ALTER
7.4 CLUSTER
7.5 COPY
7.6 GRANT/REVOKE

Development Process

WARNING for Developers: Unfortunately this list does not contain all the information necessary for someone to start coding a feature. Some of these items might have become unnecessary since they were added --- others might be desirable but the implementation might be unclear. When selecting items listed below, be prepared to first discuss the value of the feature. **Do not assume that you can select one, code it and then expect it to be committed.** Always discuss design on Hackers list before starting to code. The flow should be:

```
Desirability -> Design -> Implement -> Test -> Review -> Commit
```

НЕ ТОТ КЛАСТЕР !

Administration

☐ Allow administrators to cancel multi-statement idle transactions

This allows locks to be released, but it is complex to report the cancellation back to the client.

- Cancelling idle in transaction state

PROFESSIONAL Postgres

PostgreSQL derived databases

A list of PostgreSQL derived forks and rebranded distributions in alphabetical order.

Name	Vendor	License	Availability	Notes
Aster Data	Teradata	Proprietary	2005-....	PostgreSQL + Map/Reduce
BDR	2ndQuadrant	BSD	2014-	PostgreSQL Multi Master, contributed actively back to Core PG
Bizgres	Greenplum	BSD	2005-2007	PostgreSQL + BI features
CitusDB	Citus Data	Proprietary	2012-	Sharding and replication across many servers with MPP [1]
Cybercluster	Cybertec	BSD	2007-2010	Clustering (pgCluster fork)
Greenplum Database	Greenplum	Apache2	2005-....	PostgreSQL + BI features (formerly known as "Bizgres MPP")
ExtenDB	ExtenDB	Proprietary	2003-2007	PostgreSQL + BI Features [2]
FUJITSU Enterprise Postgres	Fujitsu	proprietary	2006-....	Full PostgreSQL compatibility with additional functionality [3]
GresCube	NTT DATA	Proprietary	2012-....	Database appliance solution based on PostgreSQL [4]
GridSQL	EnterpriseDB	GPL	2007-2010	PostgreSQL + BI Features (formerly ExtenDB) [5]
Great Bridge PostgreSQL	Great Bridge LLC	BSD	1999-2001	PostgreSQL re-distribution
HadoopDB	Yale University	Apache License V2.0	2009-....	PostgreSQL + shared-nothing cluster + Hadoop [6]
Hadapt	Teradata	Proprietary	2011-....	HadoopDB fork
Mammoth	Command Prompt	BSD	2005-2010	PostgreSQL + proprietary replication + extensions
Netezza	IBM	proprietary	2002-....	Appliance based on PostgreSQL SQL engine
NuSphere UltraSQL	NuSphere	proprietary	2002-2003	Native Win32 port of PostgreSQL
ParAccel	Actian	proprietary	2005-....	PostgreSQL + BI features [7]
Pervasive PostgreSQL	Pervasive	BSD	2005-2006	PostgreSQL re-distribution
pgCluster	SRA	BSD	2002-2005	Clustering (Share Nothing)
pgCluster-II	SRA	BSD	2006-2007	Clustering (Shared Disk)
pgPool-II	pgPool GDG	BSD	2006-....	Clustering (Connection Pooling / Replication / Load-Balancing)
PipelineDB	PipelineDB	GPL v3	2015-....	Streaming SQL
PostgresForest	NTT DATA	BSD	2006-2010	Clustering / PostgresForest is a fork of the JDBC driver, not from the backend code.
Postgres Plus	EnterpriseDB	OSS	2008-....	PostgreSQL + contrib + apps + drivers
Postgres Plus Advanced Server	EnterpriseDB	proprietary	2008-....	Postgres + Oracle compatibility + apps, formally EnterpriseDB AS [8]
Postgres-R	PGDG	BSD	2006-2010	Clustering
Postgres-X2	PGX2DG	BSD	2015-	Clustering (formerly Postgres-XC)
Postgres-XC	PGXCDG	BSD	2010-2013	Clustering [9]
Postgres-XL	PGXLDG	BSD	2014-....	Clustering
PowerGres	SRA OSS	proprietary	2003-....	Native Win32 port of PostgreSQL and Linux RPM
PowerGres Plus	SRA OSS	proprietary	2003-....	PostgreSQL + custom storage engine, redundant WAL, encrypted database [10]
PostgreSQL for Solaris	Sun	TPL	2006-2009	PostgreSQL re-distribution
RecDB	umn.edu	BSD	2013-....	Recommendation Engine [11]
Red Hat Database	Red Hat	BSD	2002-2003	PostgreSQL re-distribution
Redshift	Amazon	Private/Cloud-based	2013-....	Data Warehouse on AWS (based on ParACCEL) [12] [13]
Stado	Stado GDG	GPL	2011-2011	PostgreSQL + BI Features (fork of GridSQL) [14]
TelegraphCQ	UC Berkeley	BSD	2000-2008	Data Stream oriented fork of PostgreSQL
TruCQ	Truviso	proprietary	2008-2012	Fork of TelegraphCQ
Vertica	HP	proprietary	2005-....	Column-oriented DataWarehouse (created by Stonebraker), may only be forking the psql client library.
Yahoo! Everest	Yahoo!	private	2008-....	multi-petabyte database / MPP [15]

Кластер — важнейшая фича почти всех форков !

Политические игры

Энтерпрайз хочет энтерпрайзных решений

- ~~No~~ Cluster in **community** roadmap !

Кластерные решения сбоку:

- XC→X2, XL — Huawei, 2ndQuadrant, NTT (9.5+)
- MPPDB (XC?) — Huawei (?)
- pg_shard — CitusDB (9.5+)
- FDW — EDB+NTT (9.5+)
- Greenplum — Pivotal (8.2)
- DTM — Postgres Professional

Есть ресурсы, а решения нет :(

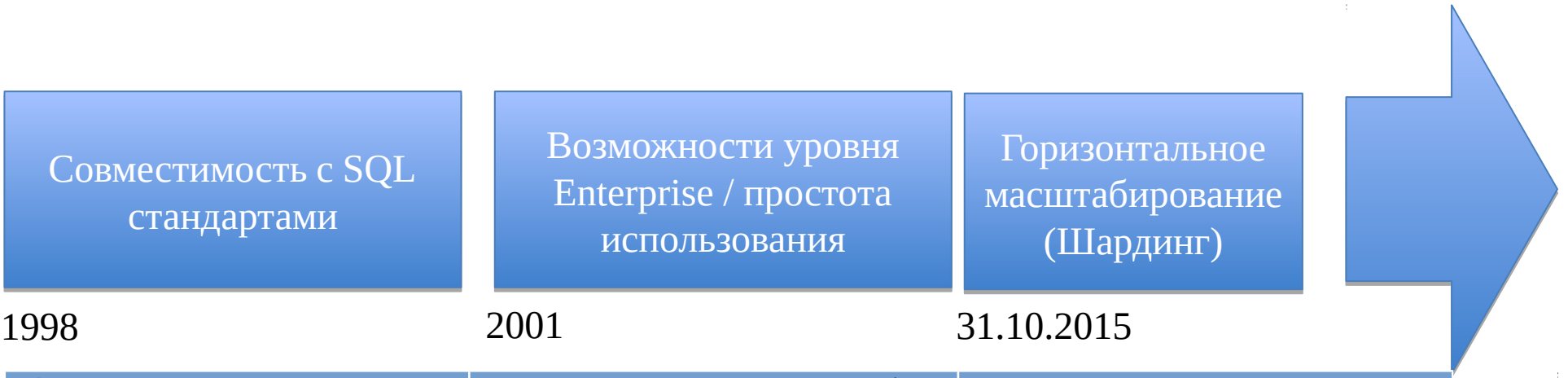
Postgres PROFESSIONAL

Cluster Summit 2015, Vienna

Cluster Summit 2015, Vienna

- Community Todo:
  - добавить КЛАСТЕР
- Продолжать развивать кластерные решения
  - Не мешать друг другу
  - Дружественность расширениям (кластерное решение как расширение ?)
- Решить с DTM для 9.6

Этапы развития PostgreSQL



Стандарты
- SQL 92 Joins
- Prepared queries
- Foreign Keys

Функциональность ядра
- Write Ahead Log
- Prepared Queries
- Info. Schema
- Auto Vacuum

Возможности уровня Enterprise
- Потоковая репликация
- Производительность
- Вертикальное масштабирование
- PITR
- pg_upgrade
- NoSQL
- BDR
- Параллелизм

Простота использования
- Портирование на Windows
- pg_basebackup
- Различные инструменты

Цели
- Read-Write scalability
- High availability

Задачи
- Управление распределенными транзакциями
- Планировщик и исполнитель распределенных запросов
- Обнаружение распределенных взаимных блокировок
- Онлайн перераспределение данных
- Поддержка глобальных ограничений
- Онлайн добавление и удаление узлов
- Средства администрирования

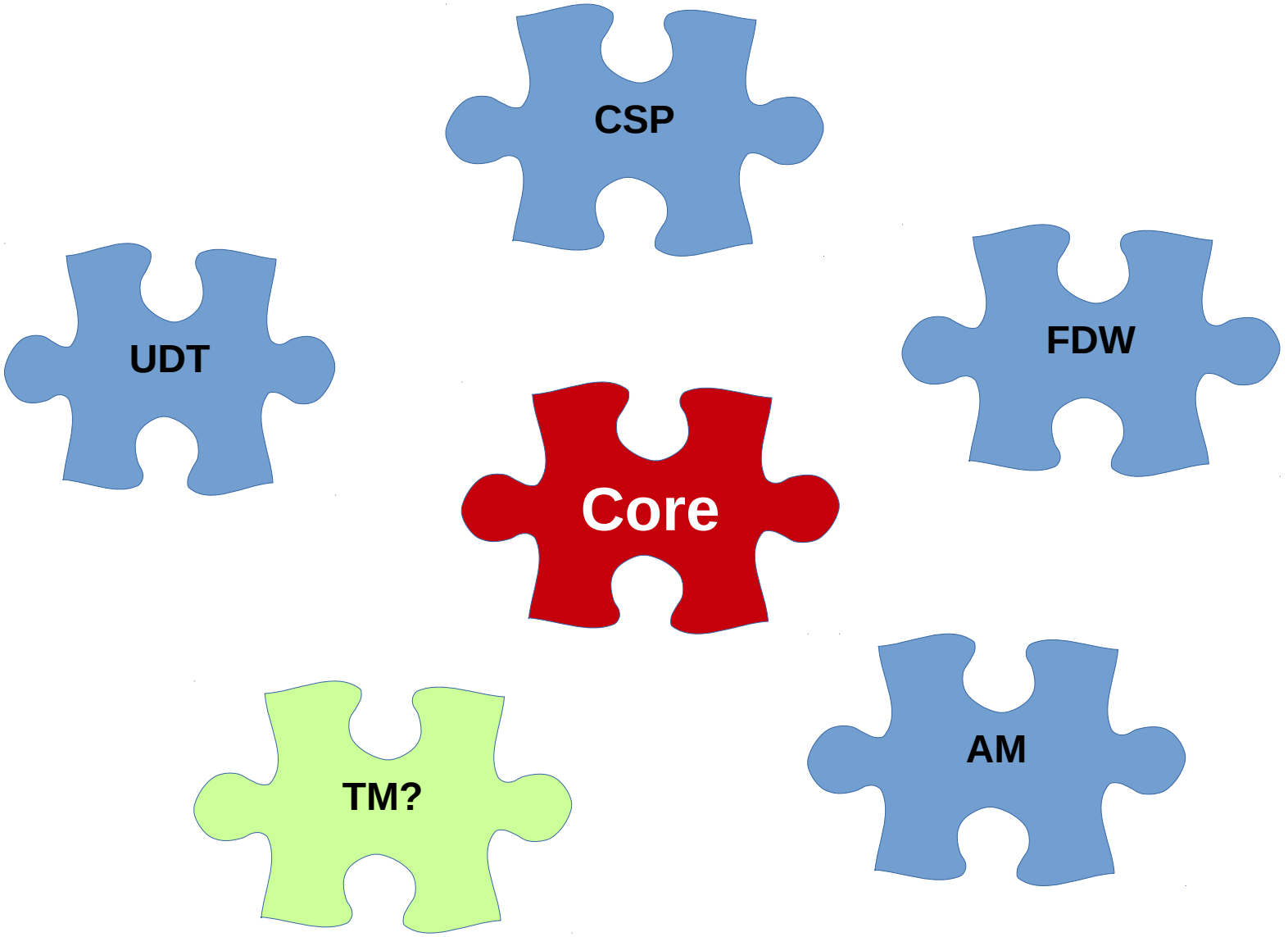
Приготовление кластера

Postgres Cluster Matrix

	WS	RS	Parallel Read	M-M	Synchr	Recov.	HA	Consistency	
								ACID	BASE
Postgres-R		+		+	+	+	+	+	
XC/XL/X2	+?	+	+	+	+			+	
PGCluster		+		+					
PgPool		+		+			+		
Pl/proxy	+	+							+
pg_shard/CituDB	+	+	+				+		
Greenplum	+	+	+						
Bucardo		+		+		+	+		+
BDR		+		+		+	+		+
SR		+			+?	+	+	+	
FDW	+	+	+		+				

Distributed transaction manager

Pluggable transaction API



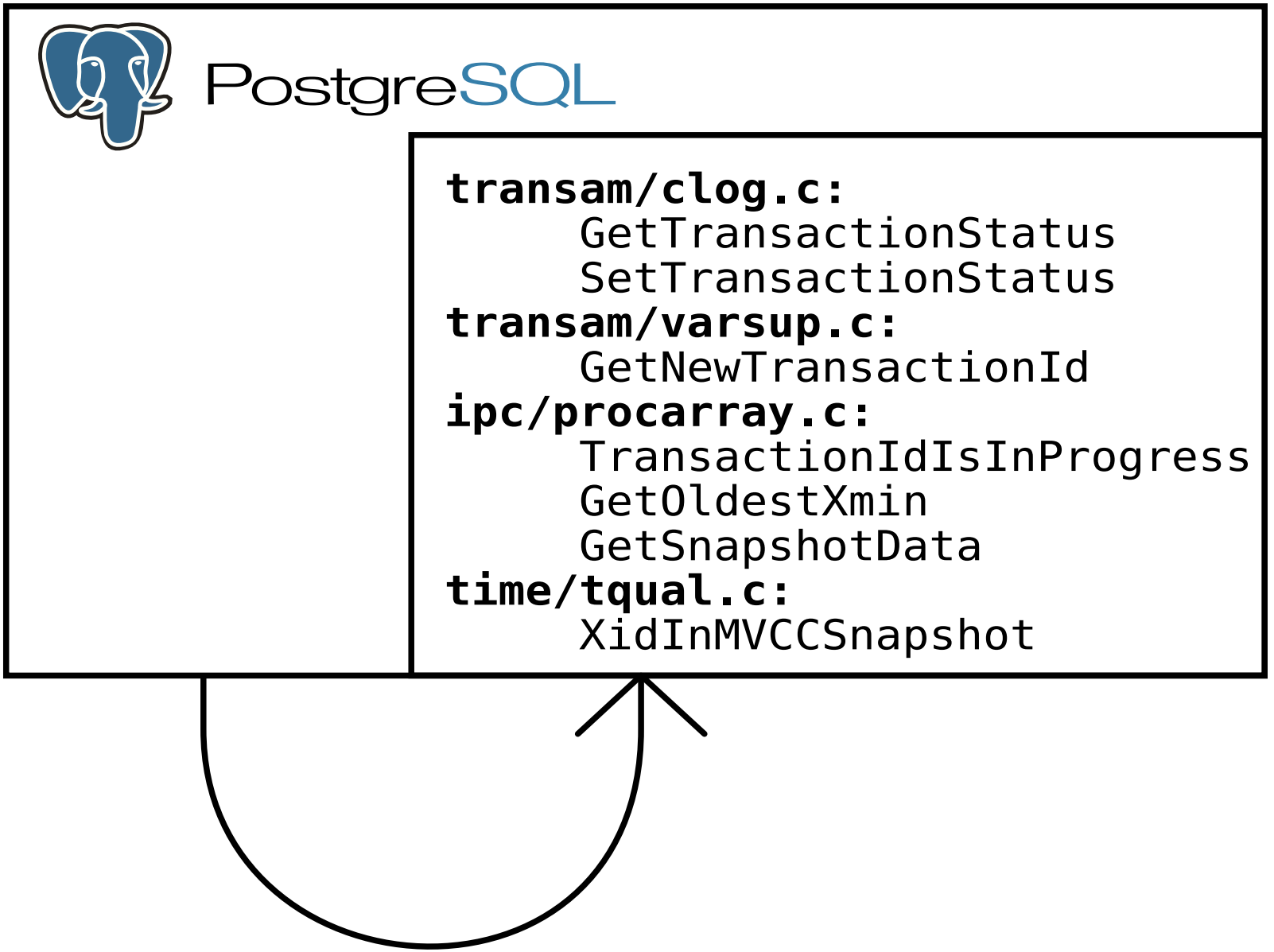
eXtensible Transaction API

- XidStatus (*GetTransactionStatus)(TransactionId xid, XLogRecPtr *lsn);
- void (*SetTransactionStatus)(TransactionId xid, int nsubxids, TransactionId *subxids, XidStatus status, XLogRecPtr lsn);
- Snapshot (*GetSnapshot)(Snapshot snapshot);
- TransactionId (*GetNewTransactionId)(bool isSubXact);
- TransactionId (*GetOldestXmin)(Relation rel, bool ignoreVacuum);
- bool (*IsInProgress)(TransactionId xid);
- TransactionId (*GetGlobalTransactionId)(void);
- bool (*IsInSnapshot)(TransactionId xid, Snapshot snapshot);

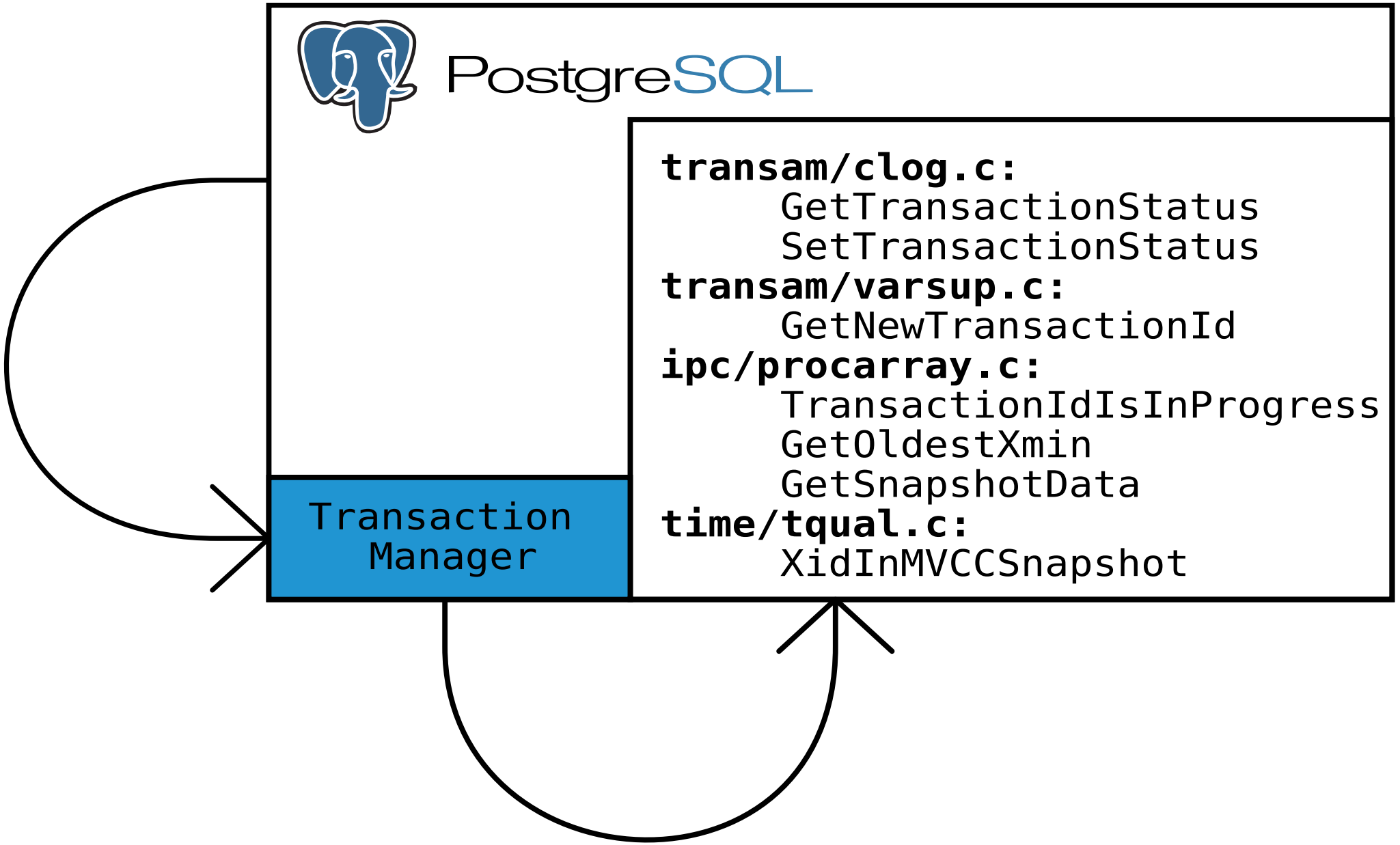
New commit callback events

- XACT_EVENT_START,
- XACT_EVENT_COMMIT,
- XACT_EVENT_PARALLEL_COMMIT,
- XACT_EVENT_ABORT,
- XACT_EVENT_PARALLEL_ABORT,
- XACT_EVENT_PREPARE,
- XACT_EVENT_PRE_COMMIT,
- XACT_EVENT_PARALLEL_PRE_COMMIT,
- XACT_EVENT_PRE_PREPARE,
- XACT_EVENT_COMMIT_PREPARED,
- XACT_EVENT_ABORT_PREPARED

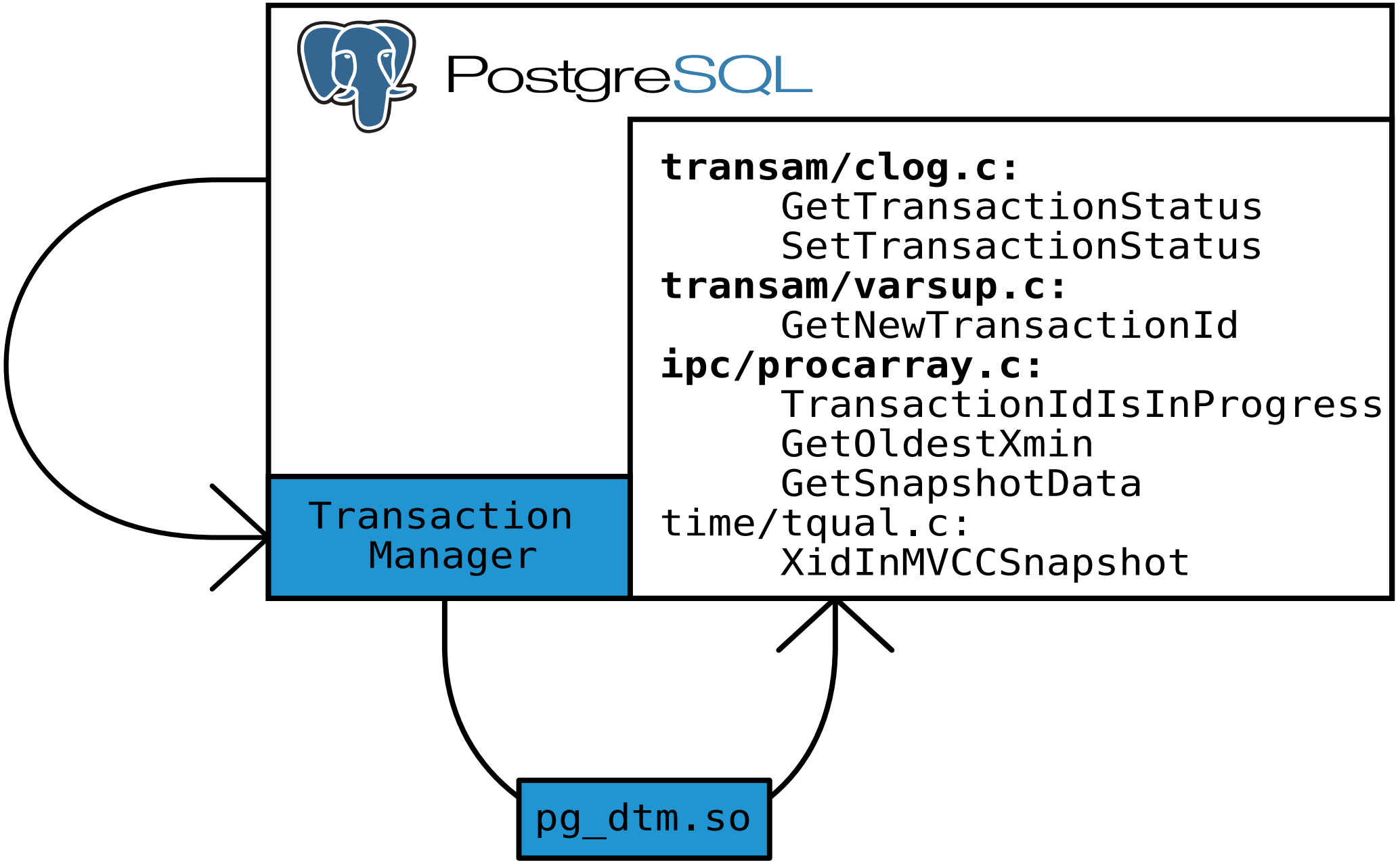
Transaction Manager before patch



Transaction Manager after patch



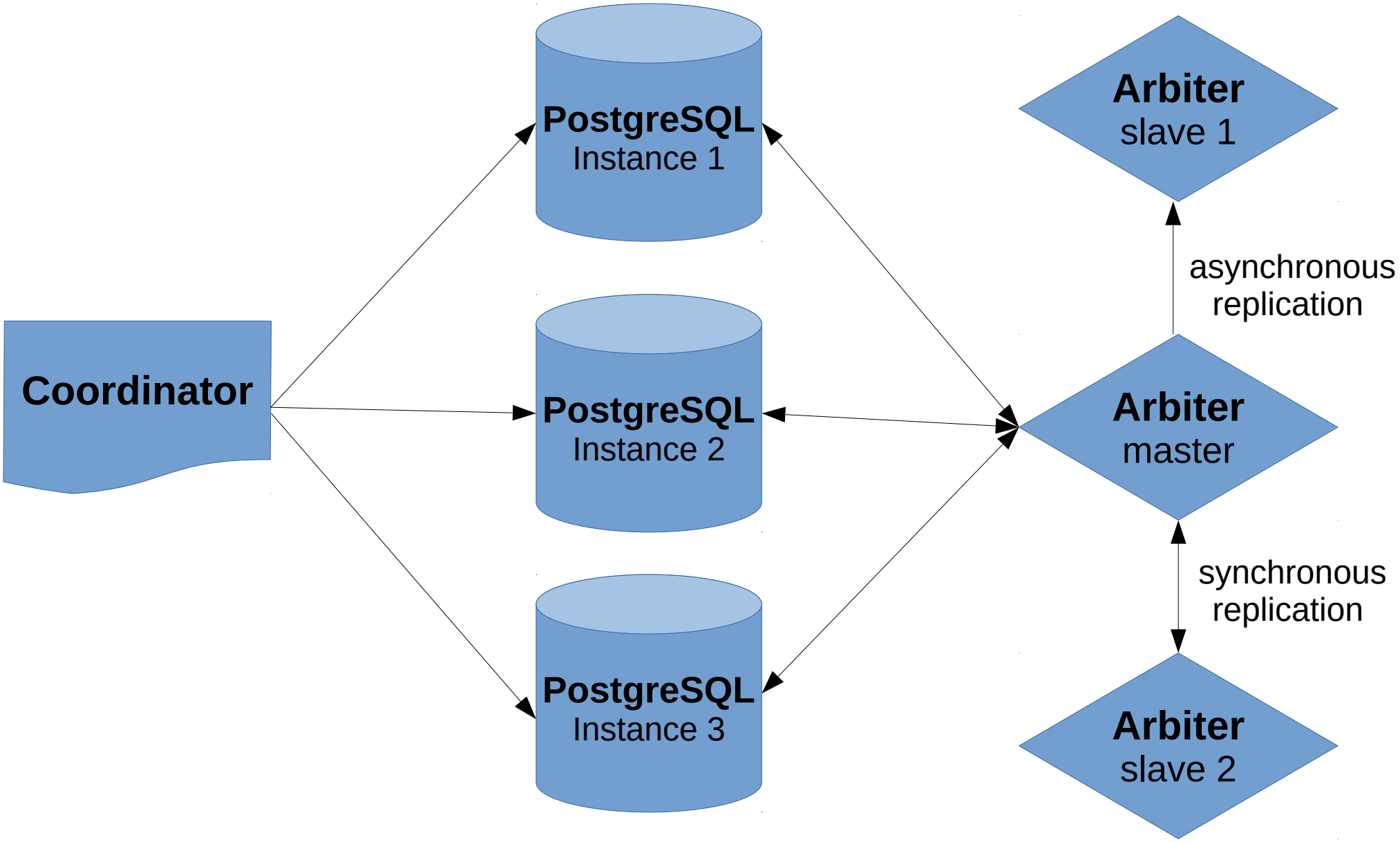
Distributed Transaction Manager



Different DTM implementations

	Local transactions	2PC	Arbiter	Examples
Snapshot sharing			✓	XL, DTM
Timestamp	✓	✓		Spanner, Cockroach, tsDTM
Incremental	✓		✓	SAP HANA

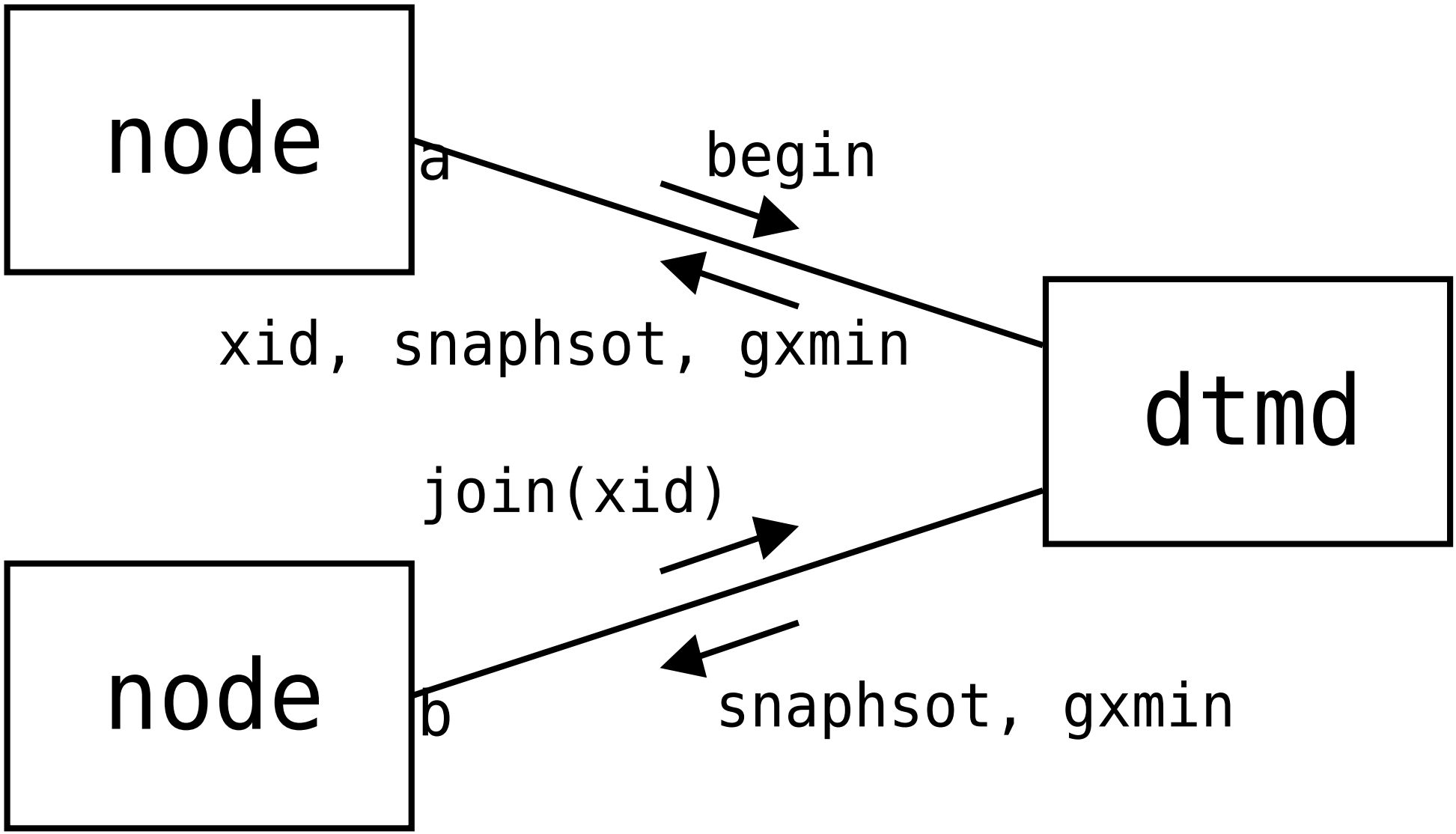
DTM architecture



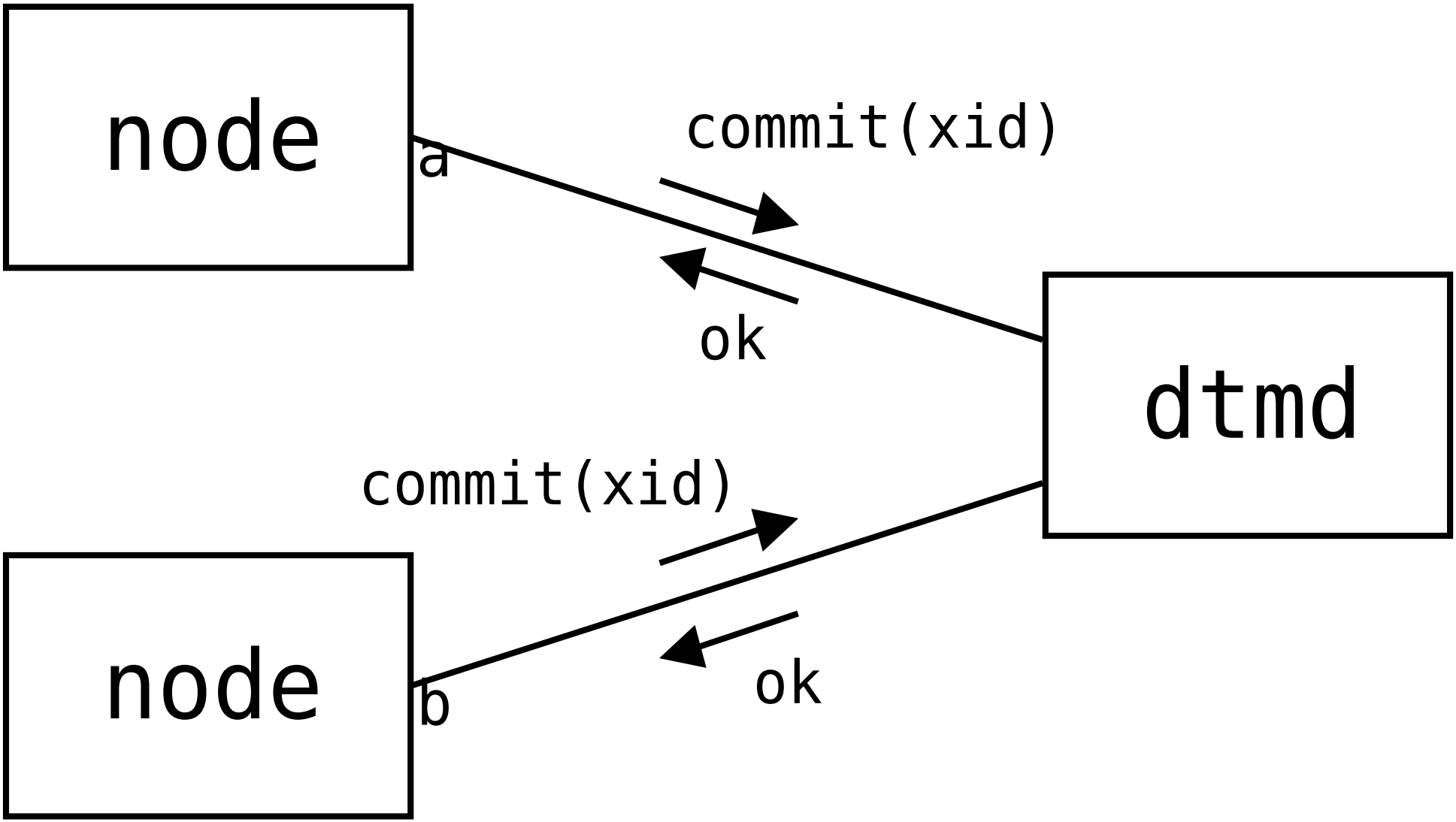
DTM from client's point of view

Primary server	Secondary server
create extension pg_dtm;	create extension pg_dtm;
select dtm_begin_transaction(); begin transaction; update...; commit;	select dtm_join_transaction(xid); begin transaction; update...; commit;

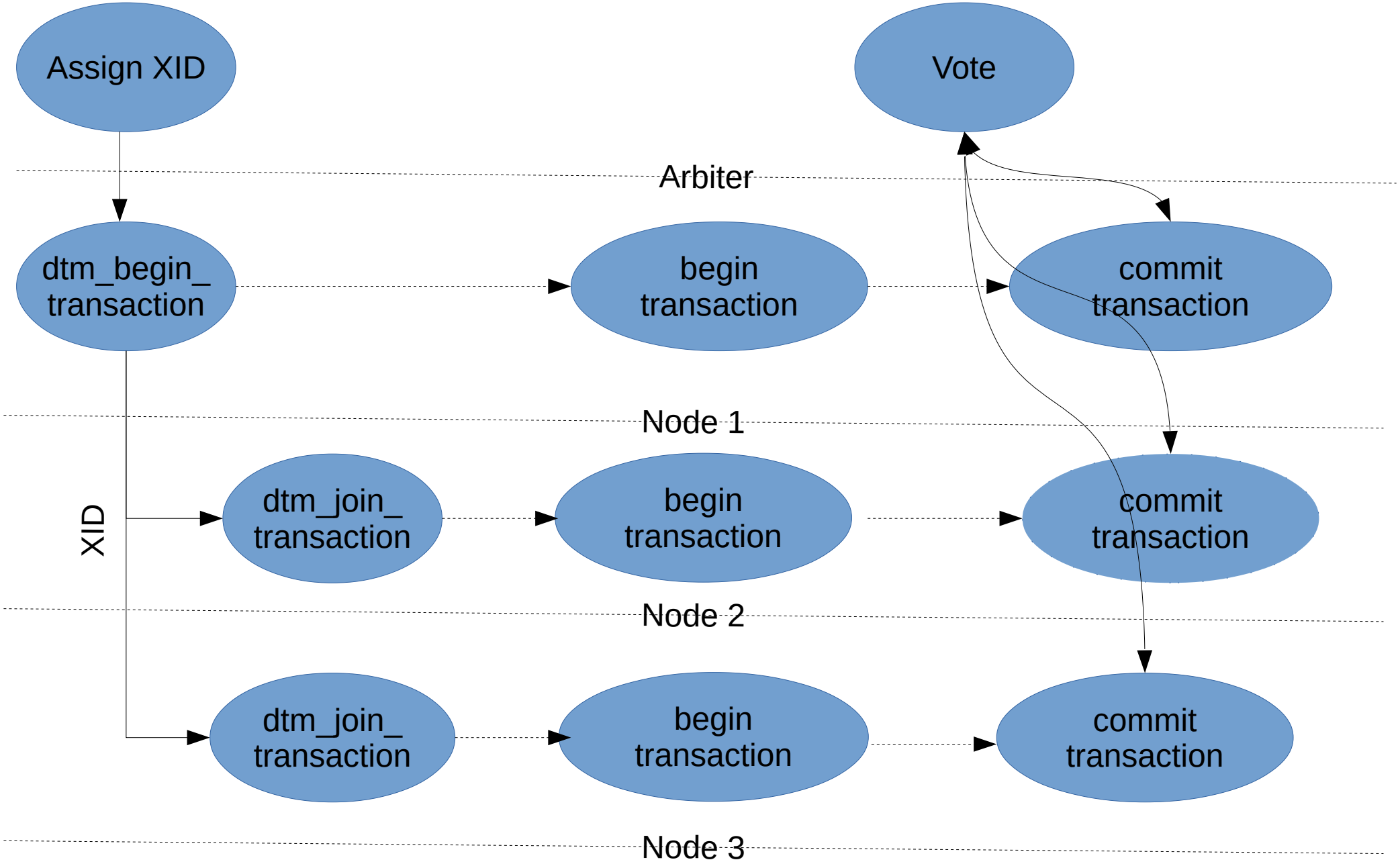
Arbiter protocol (begin)



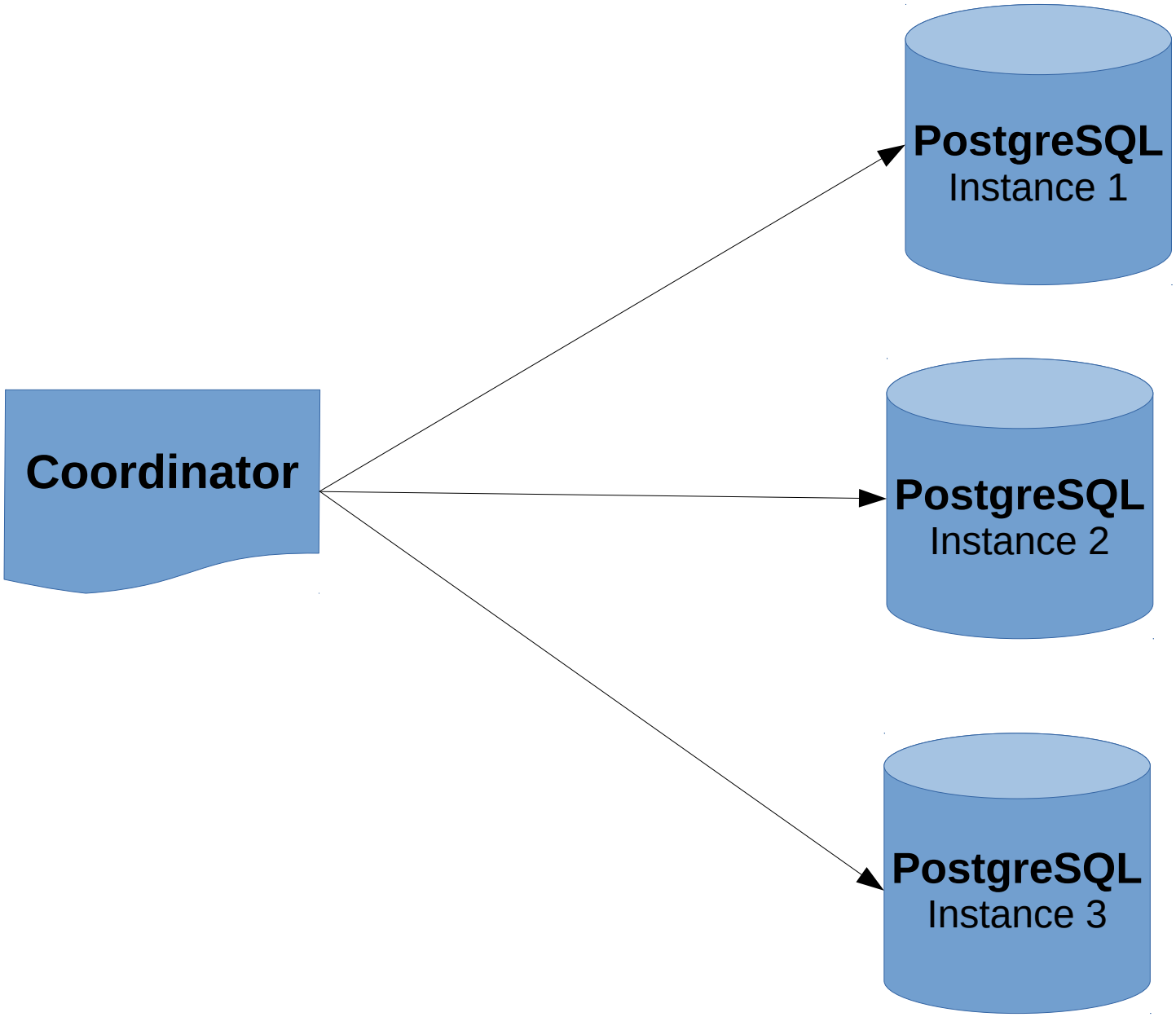
Arbiter protocol (end)



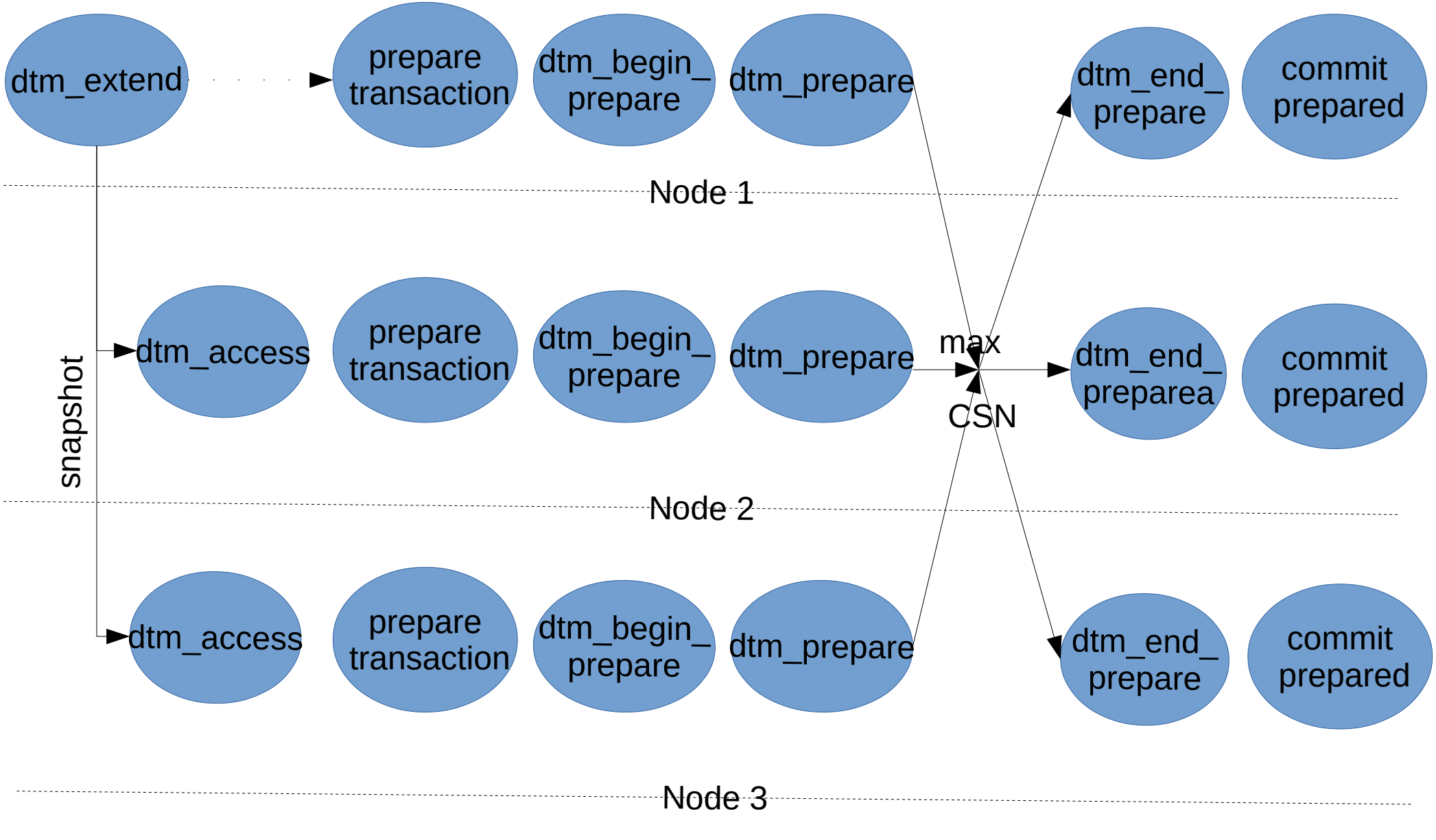
DTM transaction control flow



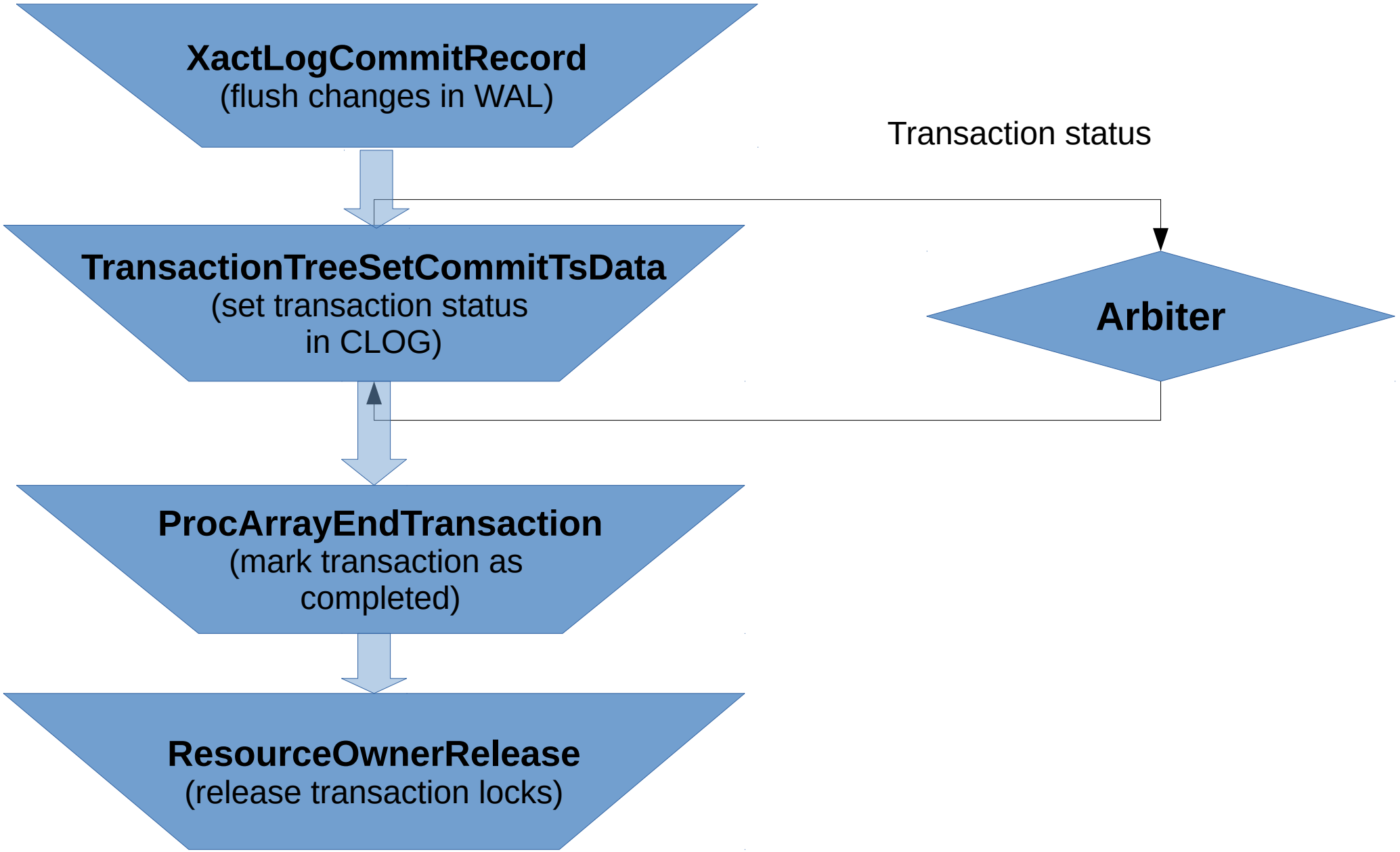
tsDTM architecture



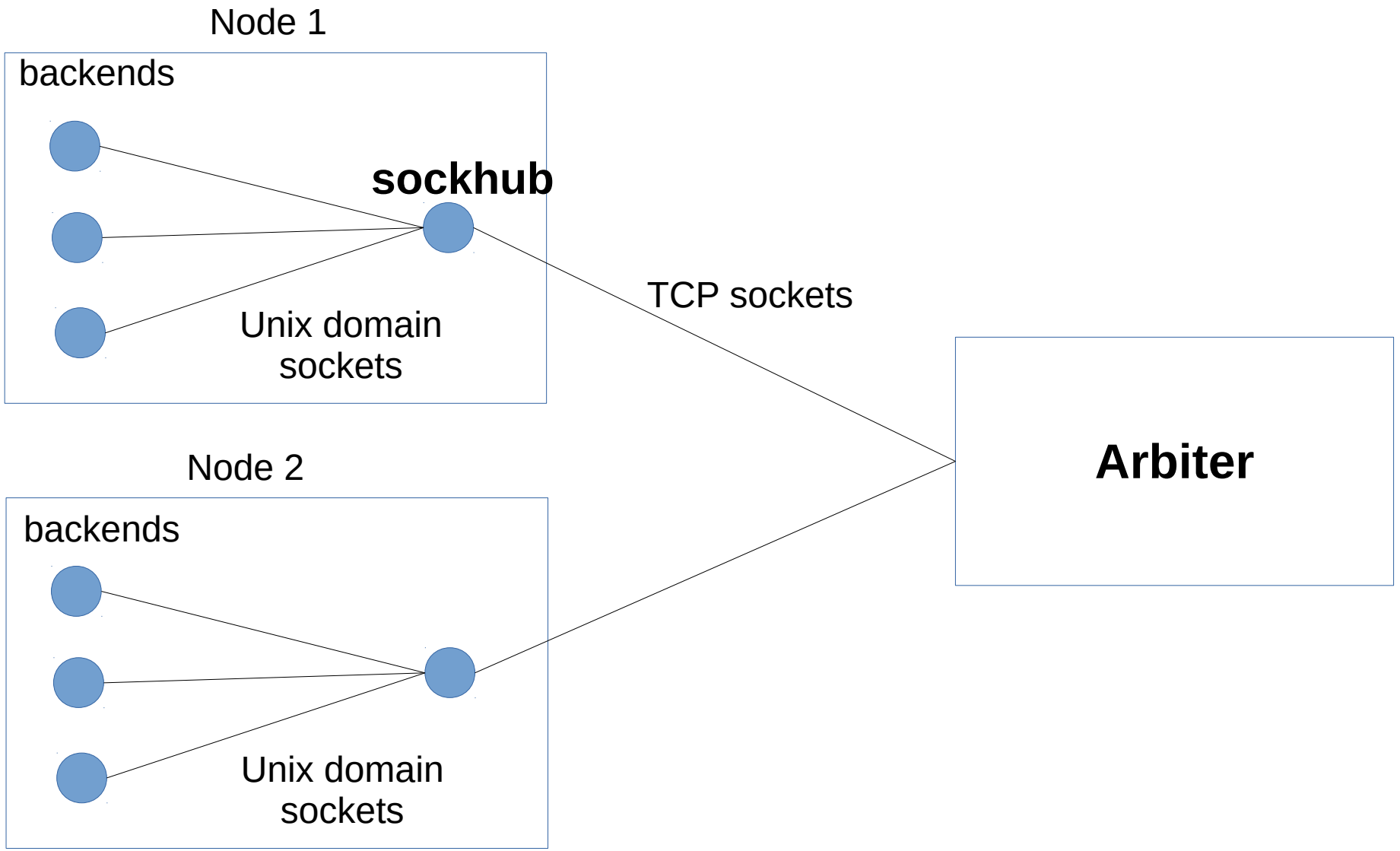
tsDTM transaction control flow



Lightweight two-phase commit



Multiplexing



Example of interaction with DTM

```
xid := execQuery(con1, "select dtm_begin_transaction()")
exec(con2, "select dtm_join_transaction($1)", xid)
exec(con1, "begin transaction")
exec(con2, "begin transaction")
exec(con1, "update t set v = v + $1 where u=$2", amount, account1)
exec(con2, "update t set v = v - $1 where u=$2", amount, account2)
var wg sync.WaitGroup
wg.Add(2)
asyncExec(con1, “commit”, &wg)
asyncExec(cnn2, “commit”, &wg)
wg.Wait()
```

Example of interaction with tsDTM

```
exec(con1, "begin transaction")
exec(con2, "begin transaction")
snapshot = execQuery(con1, "select dtm_extend($1)", gtid)
snapshot = execQuery(con2, "select dtm_access($1, $2)", snapshot, gtid)
exec(con1, "update t set v = v + $1 where u=$2", amount, account1)
exec(con2, "update t set v = v - $1 where u=$2", amount, account2)
exec(con1, "prepare transaction ''' + gtid + '''")
exec(con2, "prepare transaction ''' + gtid + '''")
exec(con1, "select dtm_begin_prepare($1)", gtid)
exec(con2, "select dtm_begin_prepare($1)", gtid)
csn = execQuery(con1, "select dtm_prepare($1, 0)", gtid)
csn = execQuery(con2, "select dtm_prepare($1, $2)", gtid, csn)
exec(con1, "select dtm_end_prepare($1, $2)", gtid, csn)
exec(con2, "select dtm_end_prepare($1, $2)", gtid, csn)
exec(con1, "commit prepared ''' + gtid + '''")
exec(con2, "commit prepared ''' + gtid + '''")
```

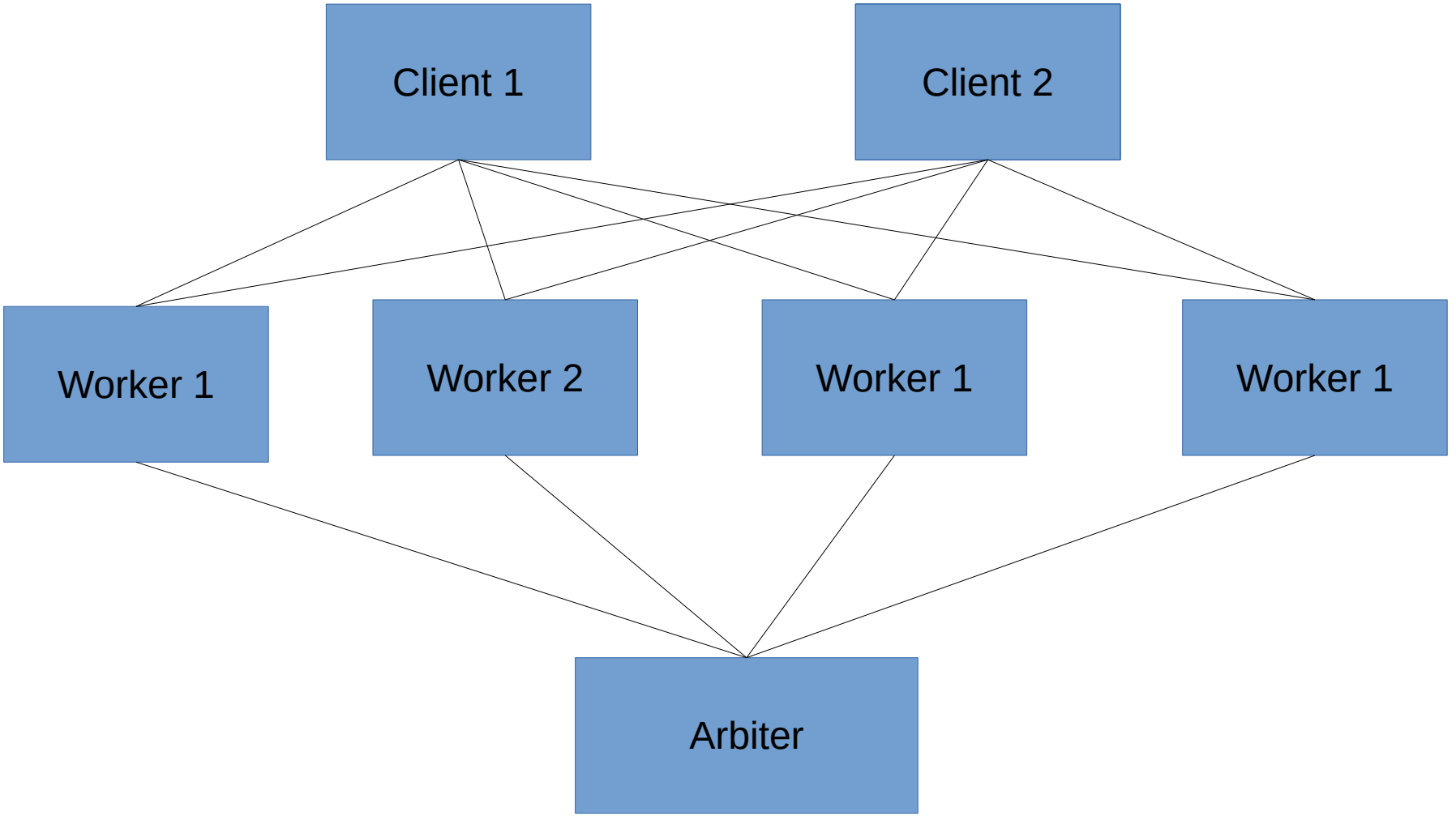
Example of using FDW

```
exec(con, "select dtm_begin_transaction()")
exec(con, "begin transaction")
exec(con, "update t set v = v + $1 where u=$2",
        amount, account1)
exec(con, "update t set v = v - $1 where u=$2",
        amount, account2)
exec(con, “commit”)
```

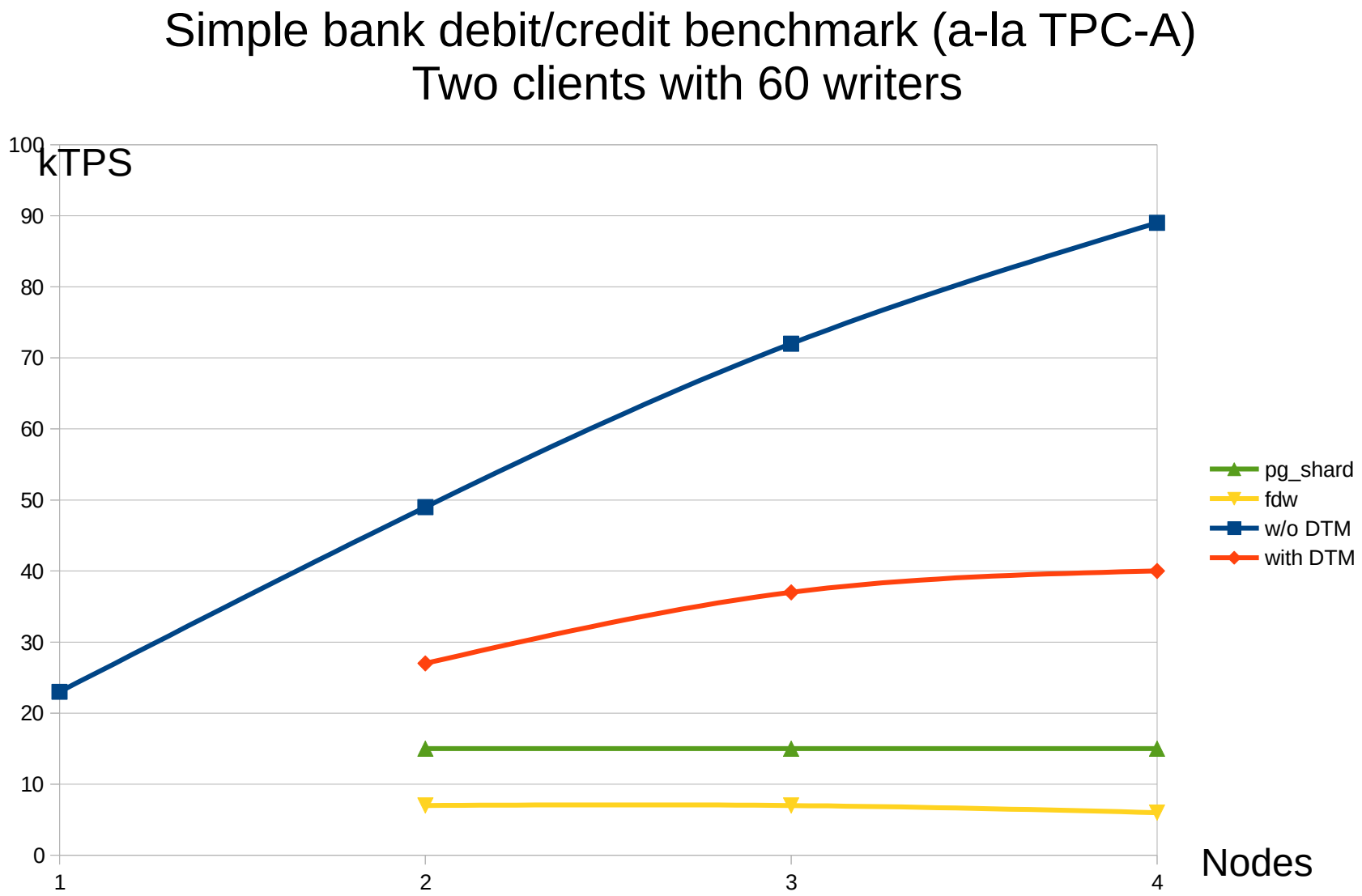
Example of using pg_shard

```
exec(con, "begin transaction")
exec(con, "update t set v = v + $1 where u=$2",
        amount, account1)
exec(con, "update t set v = v - $1 where u=$2",
         amount, account2)
exec(con, “commit”)
```

Test configuration

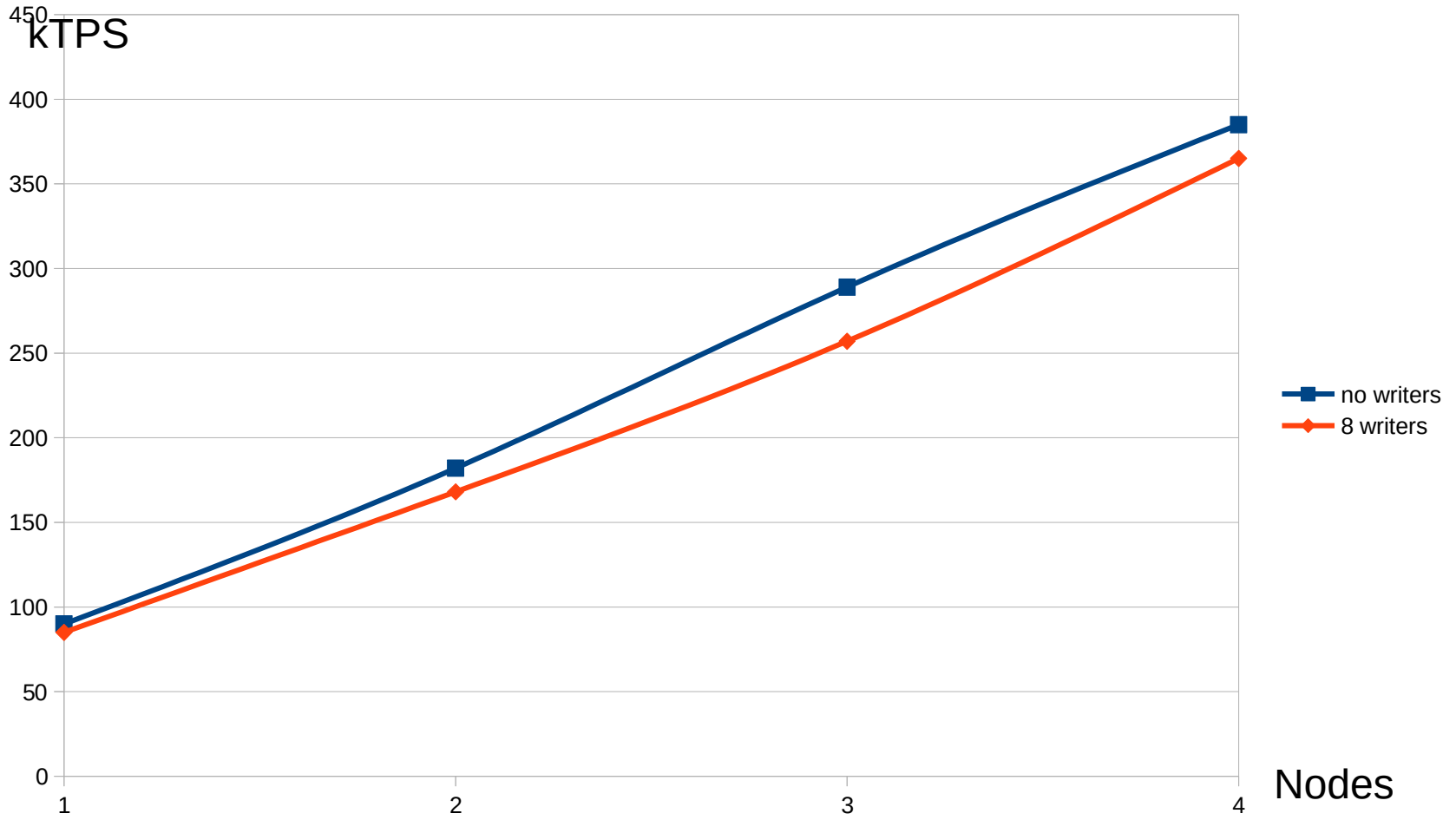


Performance measurement



Multimaster performance

Simple update/select queries
Three clients with 140 readers



Roadmap

- Add XTM patch to PostgreSQL 9.6
- Experiment with different DTM implementations
- Provide integration of DTM with different cluster solutions (pg_shard, FDW, XL,...)
- Implement multimaster on top of DTM

Postgres Cluster Matrix

	WS	RS	Parallel Read	M-M	Synchr	Recov.	HA	Consistency	
								ACID	BASE
Postgres-R		+		+	+	+	+	+	
XC/XL/X2	+?	+	+	+	+			+	
PGCluster		+		+					
PgPool		+		+			+		
Pl/proxy	+	+							+
pg_shard/Citus	+	+	+		+		+	+	
Greenplum	+	+	+						
Bucardo		+		+		+	+		+
BDR		+		+		+	+		+
SR		+			+?	+	+	+	
FDW	+	+	+		+			+	

Postgres PROFESSIONAL

Спасибо за внимание!