



# PostgreSQLの構成管理

---

日本PostgreSQLユーザー会

石井達夫

[ishii@postgresql.jp](mailto:ishii@postgresql.jp)



# PostgreSQLとは

---

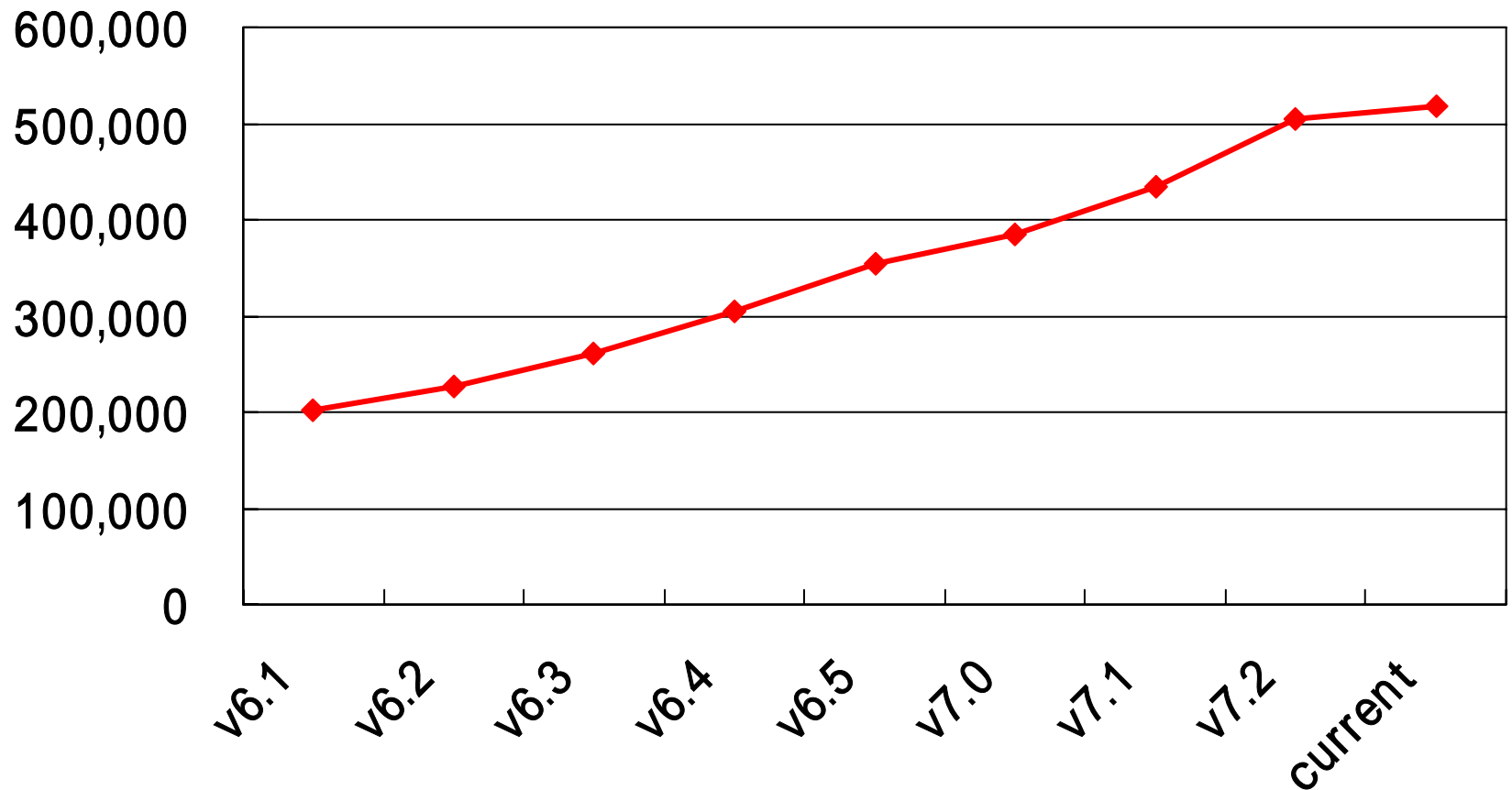
- BSD Unixで有名なカリフォルニア大学バークレー校で開発
- BSDライセンス
- 「純粋」なオープンソースデータベース
- SQL99にもとづく関係データベースの実装
- 商用DBに匹敵する機能 + オブジェクト指向的な拡張機能
- “Real World”で広く使われている



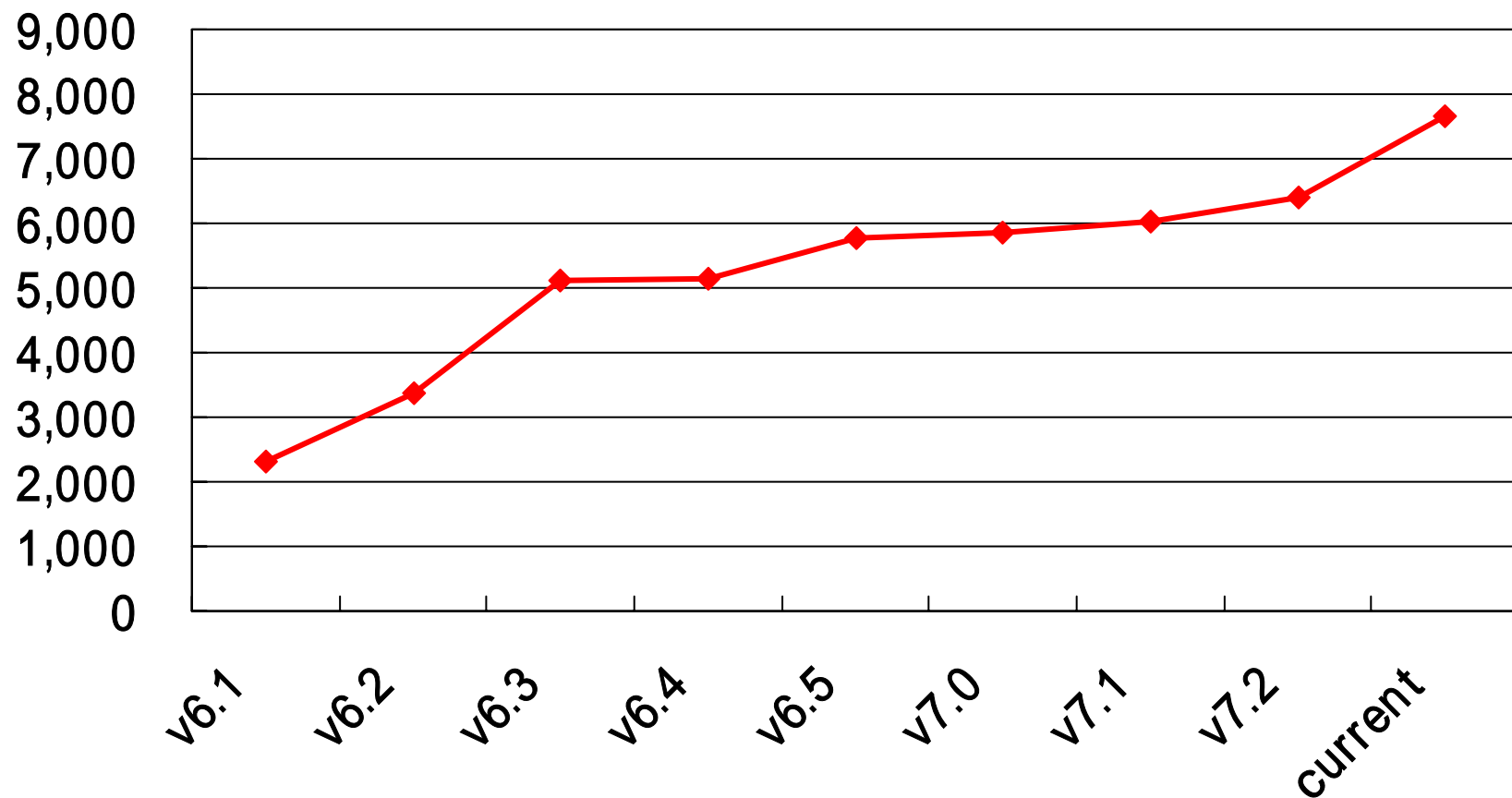
# PostgreSQLの歴史

6.1	1997/06	GEQOオプティマイザ、GROUP BY、SETコマンド
6.2	1997/10	JDBCドライバ、TRIGGER、多数のSQL92データ型
6.3	1998/03	subselect、マルチバイトサポート、ecpg
6.4	1998/10	PL/PgSQL、PL/tcl、HAVING
6.5	1999/11	行ロック、MVCC
7.0	2000/05	外部キー
7.1	2001/04	WAL、TOAST、OUTER JOIN
7.2	2002/02	Concurrent VACUUM
7.3	2003?	Schema

# ソースコード行数の変遷



# 文法定義ファイル行数の変遷





## この5年間に...

---

- ソースコード行数は2.5倍に  
(202,298→517,563)
- 文法の複雑さは3.3倍に(2,320→7,650)
- リリース期間は4～13ヶ月
  - リリースの間に数万行コードが増える



# データベース開発の特殊性 (IMO)

---

- 開発者にとってよいニュース
  - 要求定義はあらかじめできている(SQL99)
  - 実装手法も確立している(MVCC, 国際化を除く)
  - 作るだけ！
- 悪いニュース
  - 複雑、大規模
  - 安定性が重要
  - 遅いDBは誰も使ってくれない
    - 開発言語はCが今でも主力
- 「カリスマ」がない – 全員が「第2世代」



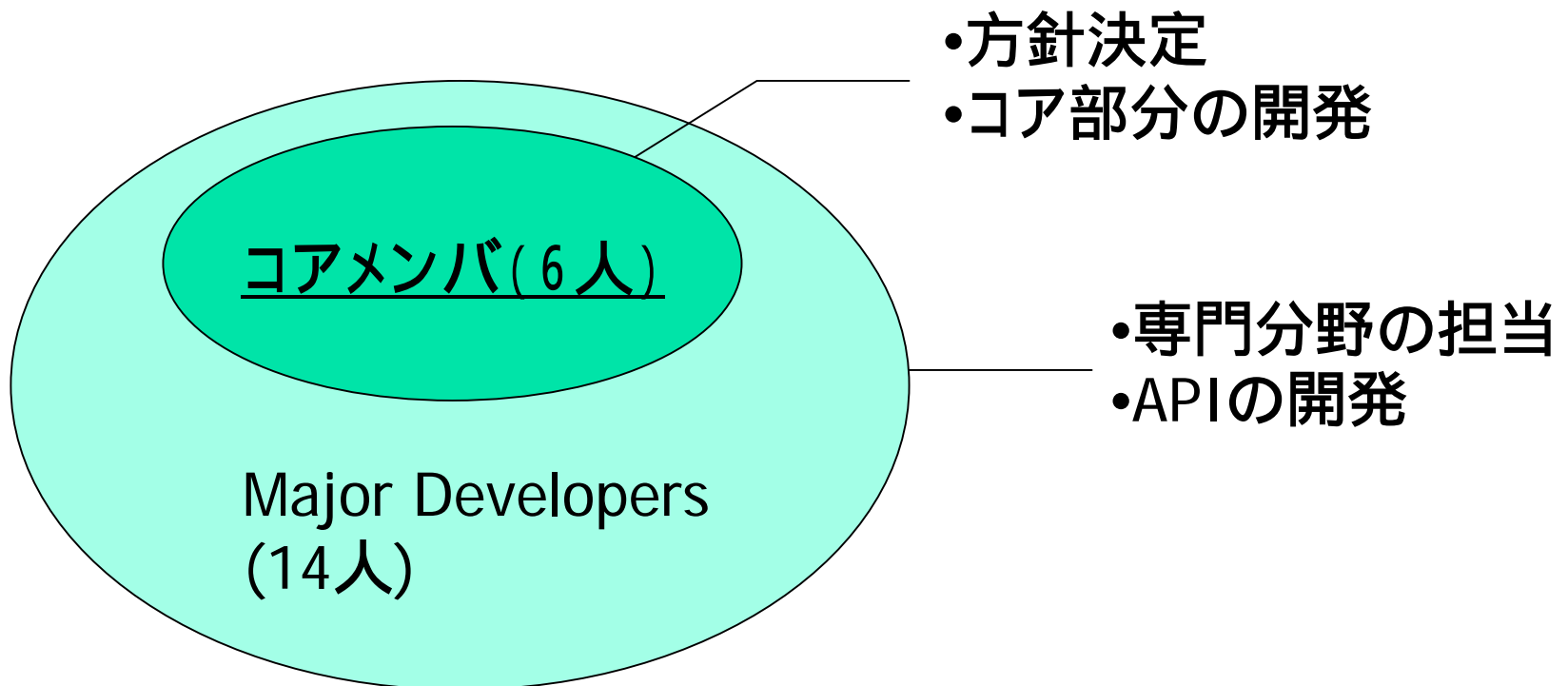
# PostgreSQL開発のゴール

---

- 目標
  - 安定性
  - コードの品質維持
  - 機能拡張
- 方針
  - 少数精鋭主義
  - 公開レビュー・議論
  - テストの徹底

# 開発体制

## ■ BSD Unixの開発をモデルとする



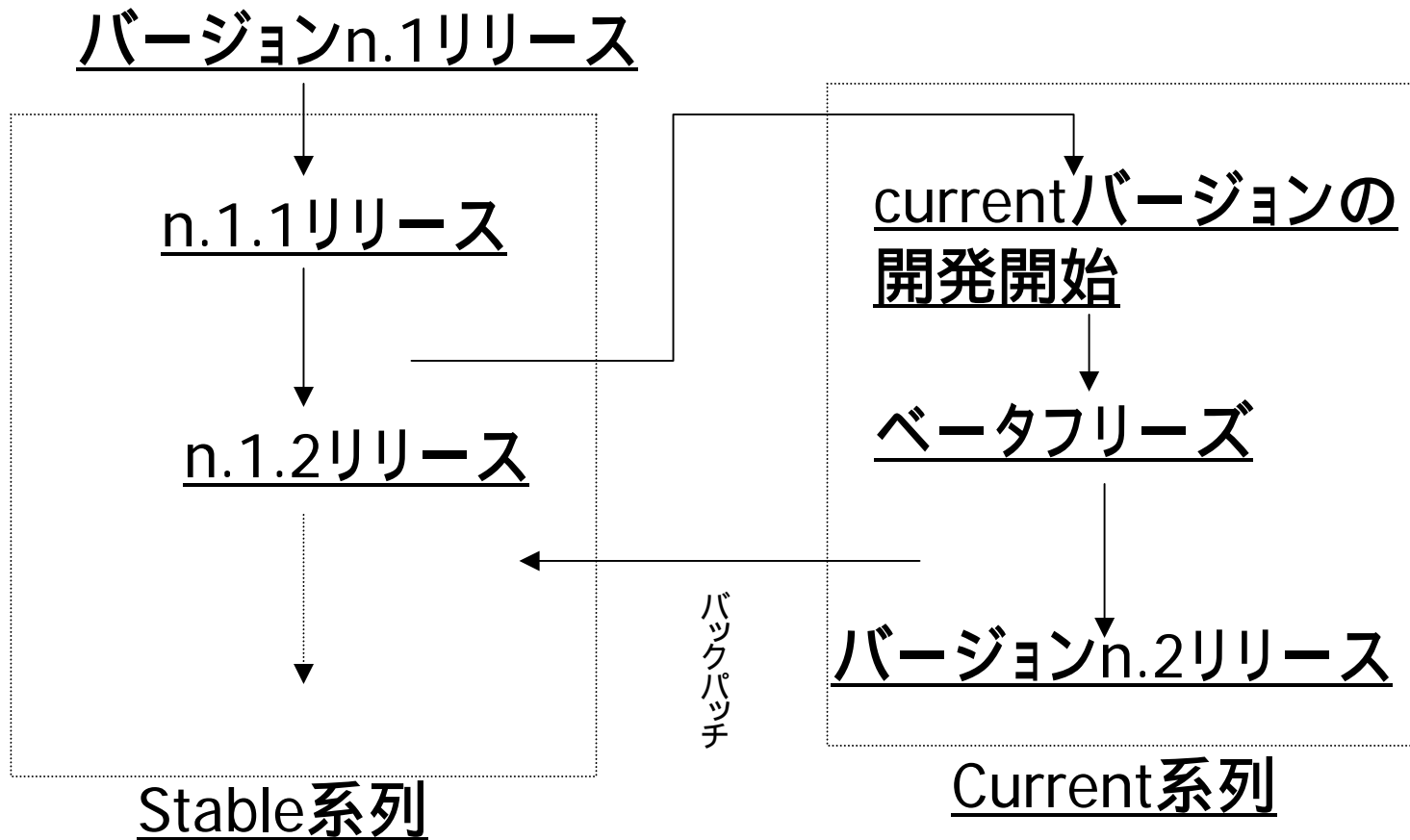


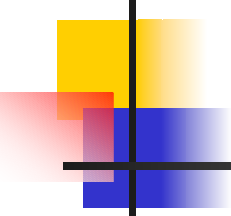
# 開発資金

---

- サーバ、回線はコアメンバーの一人が提供
- 何人かのコアメンバーは企業のスポンサードにより、フルタイムで開発に従事
- ほかはすべてボランティアベース
- 純粋なオープンソースなので、常に資金不足(sourceforgeやOSDLの協力は得ていない)

# リリースサイクル





# 「メジャーバージョンアップ」と 「マイナーバージョンアップ」

- メジャーバージョンアップ
  - n.1→n.2
  - データベースの互換性なし
  - プロトコルの互換性なし
  - 機能追加
  - current系列での開発
- マイナーバージョンアップ
  - n.1.1→n.1.2
  - データベース、プロトコルの互換性あり
  - 機能追加なし、バグ修正のみ
  - stable系列での開発



# 個々の機能の開発の流れ

---

機能追加・変更の提案



議論



実装の提案(パッチ)  
+ 回帰テスト + ドキュメント



CVSにコミット



# TODO

---

- 今後実現すべき機能、修正すべき問題点のリスト
- 開発中に頻繁に項目追加、削除(実際にはダッシュマークを付与)



# TODOの例

---

- \* Allow `elog()` to return error codes, module name, file name, linenumber, not just messages (Peter E)
- \* Add error codes (Peter E)
- \* -Change DEBUG startup tag to LOG (Bruce)
- \* Show location of syntax error in query [yacc]
- \* -Add `getpid()` function to backend
- \* Allow logging of query durations



# 開発ツール

---

- メールングリスト
  - 100通 / 1日
- CVS
  - 複数ブランチなし、マージは行わない
- 回帰テスト(regression test)
  - 1万行以上のSQL文による自動テスト
- ベンチマークテスト
- Assertion(表明)
- 複数プラットフォームでの検証



# ドキュメント管理

---

- SGML(Docbook)
- HTML/manへの自動変換
- ドキュメントの更新とプログラムの更新の同期



# 評価

---

- よい点

- コードの品質

- コアメンバなどによる少数精鋭開発により、品質を維持

- 安定性

- 回帰テスト、長いベータテスト期間により、バグ退治



# 評価(続き)

---

## ■ 悪い点

- 機能拡張のスピード
  - 周辺API、ツールの進化とリリースサイクルが合わない
- 性能評価が十分に行われていない
  - 他商用DBとのベンチマーク比較ができない(メーカーが拒んでいる)
  - 大量のリソースが必要



# 今後の方向性

---

- 独立性の高いモジュールは別サーバで開発
  - pgaccess, JDBC, ODBC...
- ベンチマークツールの開発、採用
  - 独自ツール
  - 標準ツール



# 今後の課題

---

- スポンサー
  - 特にコアメンバーのパトロン企業
- 若いエンジニアの参加
  - 教育や内部ドキュメントの充実