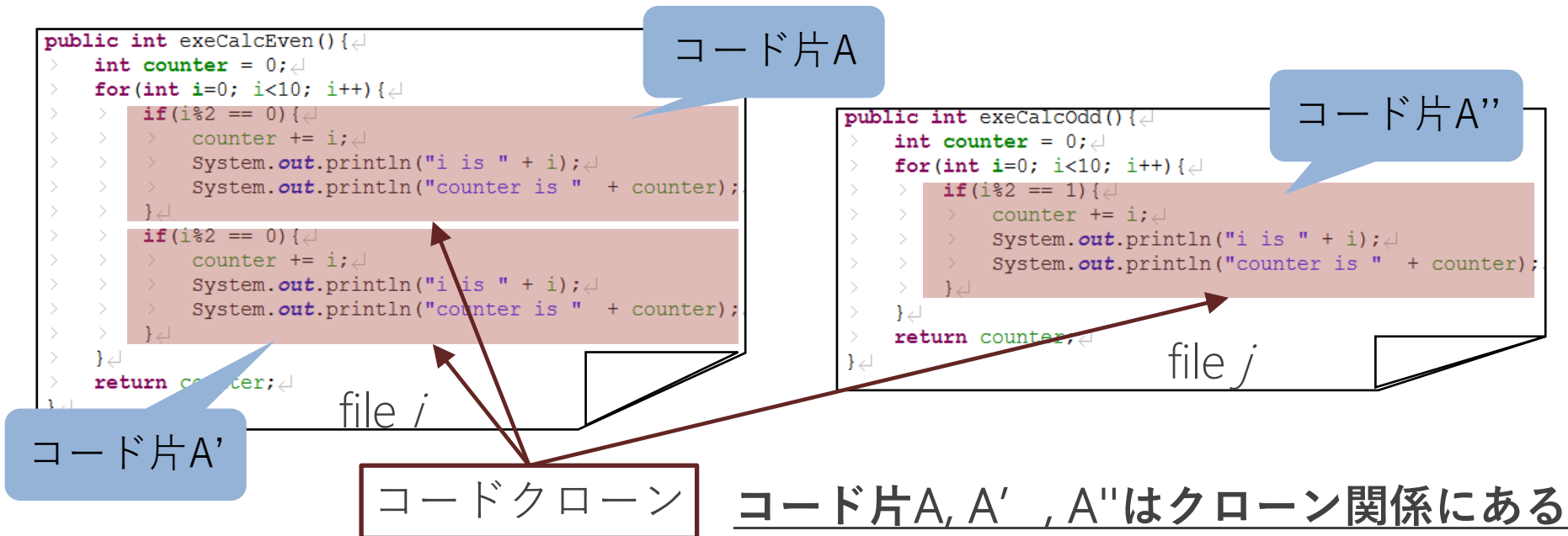


コードクローンの作成・利用過程とソフトウェア品質の関係分析

久木田 雄亮

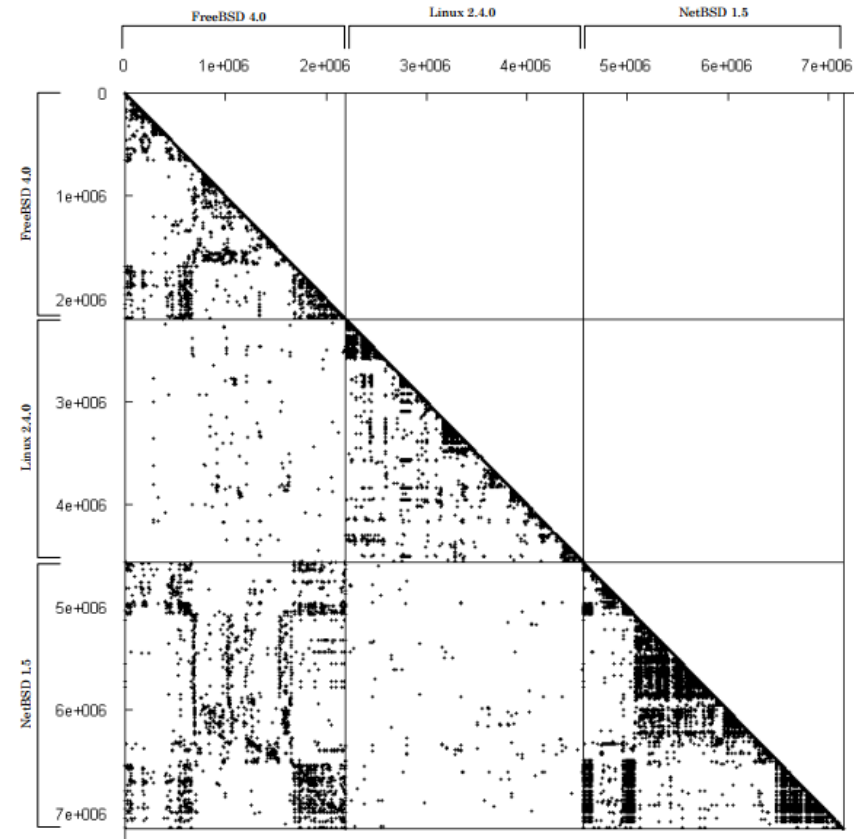
コードクローンとは

- ソースコード中に存在する「同一の」または「類似する」コード片のこと
 - 主に、既存コードを再利用（コピーアンドペースト）することにより作成される



研究の背景 (1/2)

- ソフトウェア開発・保守の過程では、生産性の向上を目的として既存コードが頻繁に再利用されることがある
- コードクローンがソースコード全体に遍在することが少なくない

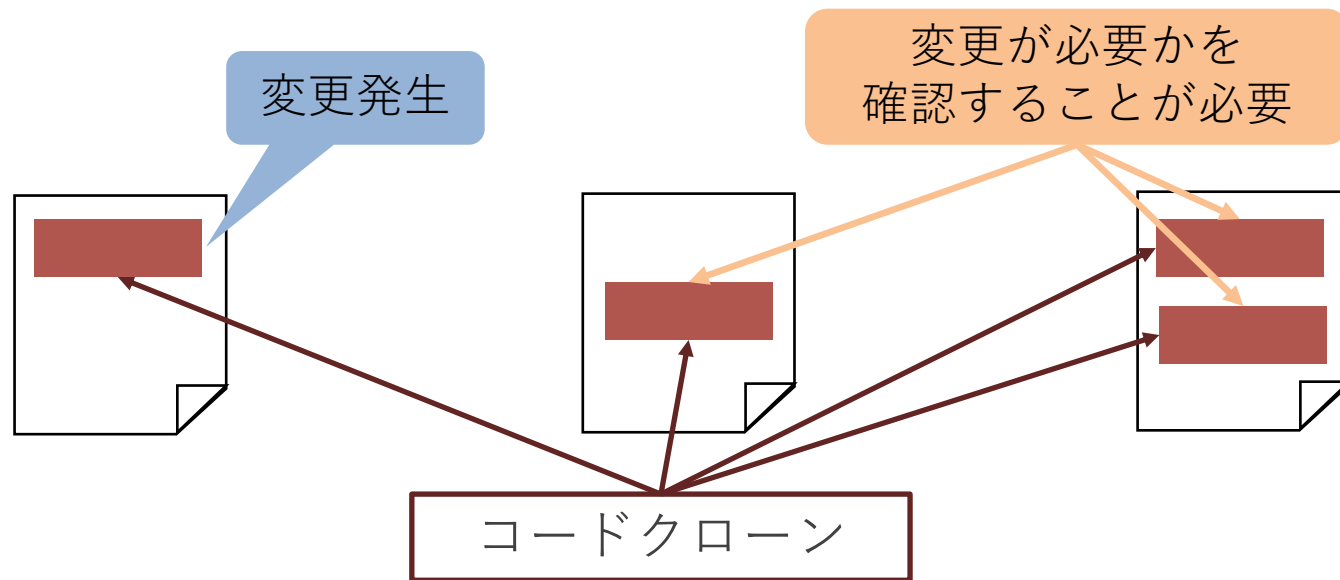


UNIX系OS, Linux間のコードクローン[1]

[1] Kamiya, T., Kusumoto S. and Inoue K.: CCFinder: A multilinguistic token-based code clone detection system for large scale source code, IEEE Transactions on Software Engineering (TSE), Vol.28, No.7, pp.654-670, 2002.

研究の背景 (2/2)

- 欠陥除去などの理由によりコードクローンに変更を加える際のリスク
 - 変更箇所が多岐に渡るため保守コストが増加する
 - 変更漏れにより新たな不具合を混入してしまう



これまでのコードクローン研究

- コードクロンの検出手法
 - 主眼：大規模なシステムのソースコードを高速に解析しコードクロンを検出すること
 - ソースコードをトークン（リテラル，括弧，演算子）の並びで表現し，一致する並びをクローンとして検出[1]
- コードクロンの分析手法
 - 主眼：コードクロンの特徴と品質（バグ発生確率）との関係を明らかにすること
 - コードクロンのタイプ（違いの度合い）ごとのバグ発生確率に関する分析[2]

[1] Kamiya, T., Kusumoto S. and Inoue K.: CCFinder: A multilinguistic token-based code clone detection system for large scale source code, IEEE Transactions on Software Engineering (TSE), Vol.28, No.7, pp.654-670, 2002.

[2] M. Mondal, C. K. Roy and K. A. Schneider, "A comparative study on the bug-proneness of different types of code clones," IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 91-100, 2015.

先行研究の問題点

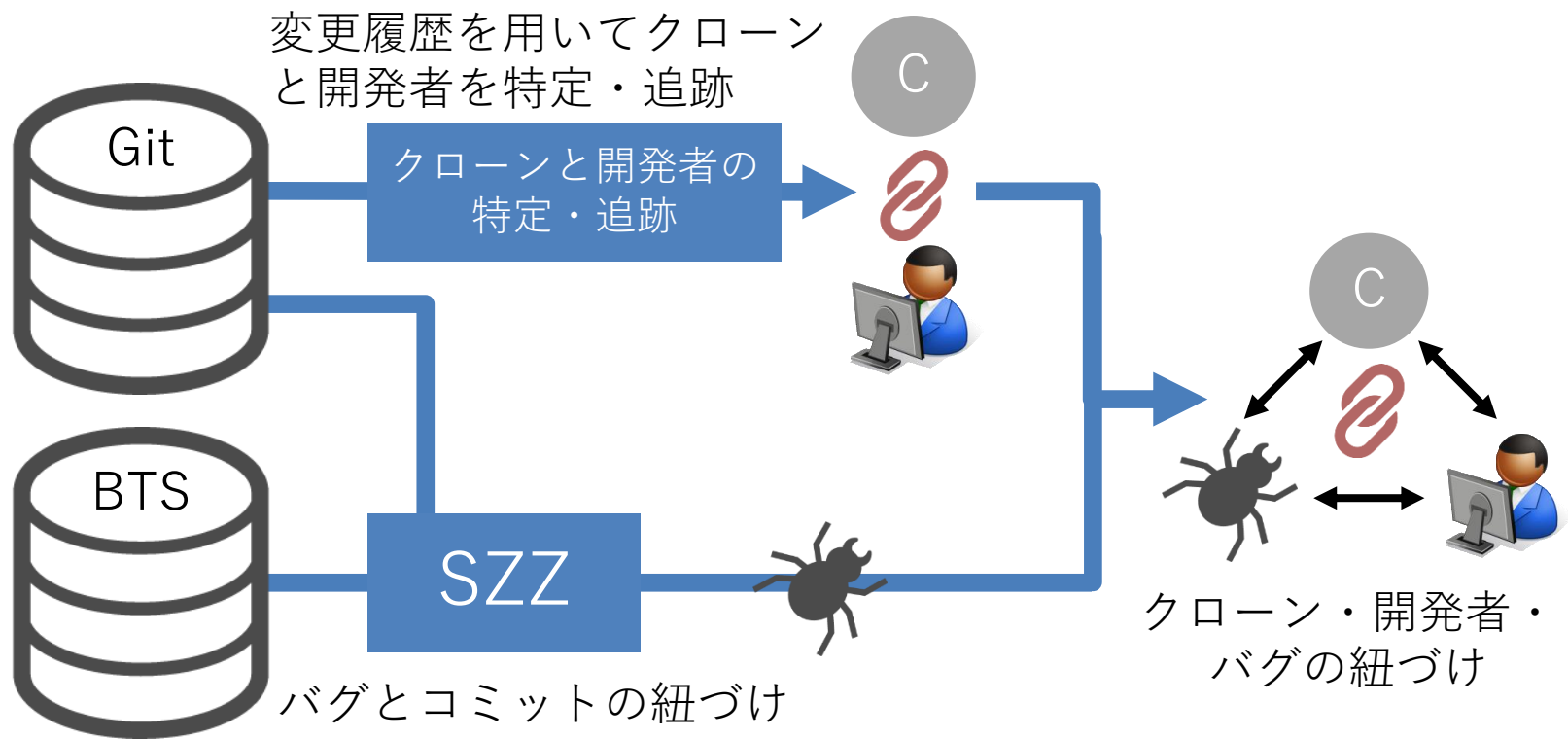
- コードクローンの作成・利用過程と品質の關係に着目した分析がおこなわれていない
 - ある時点でのコードクローンの特徴とソフトウェア品質の關係に着目したものがこれまでの研究

研究の目的

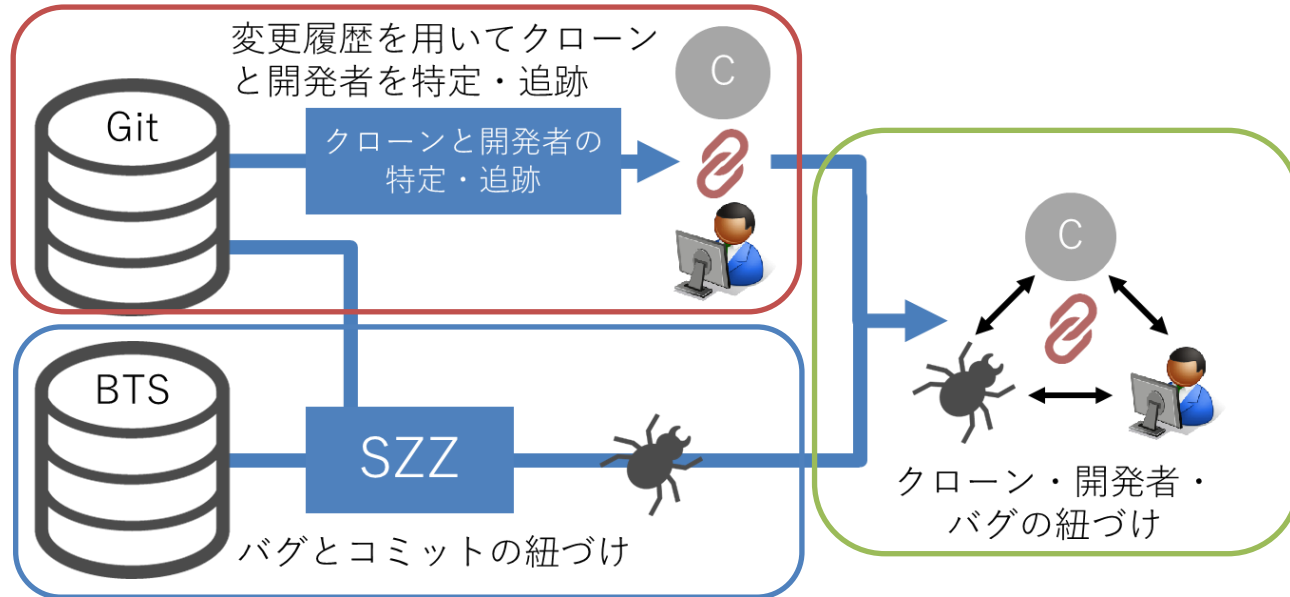
- 目的
 - コードクロンの長期的かつ安全な管理方法について、**コードクロンの作成・利用過程の観点**から有益な知見を提供すること
- アプローチ
 - コードクロンの作成・利用過程における人的影響を調査するためのツールを構築する
 - コードクロンの作成・利用過程と開発者の追跡
 - SZZアルゴリズムによるバグとクロンの紐付け
 - コードクロン作成・利用過程と品質の関係の分析

CCT: Code Clone Tracer

- コードクローンの作成・利用過程における人的影響の調査を支援するためツール



CCTの処理手順

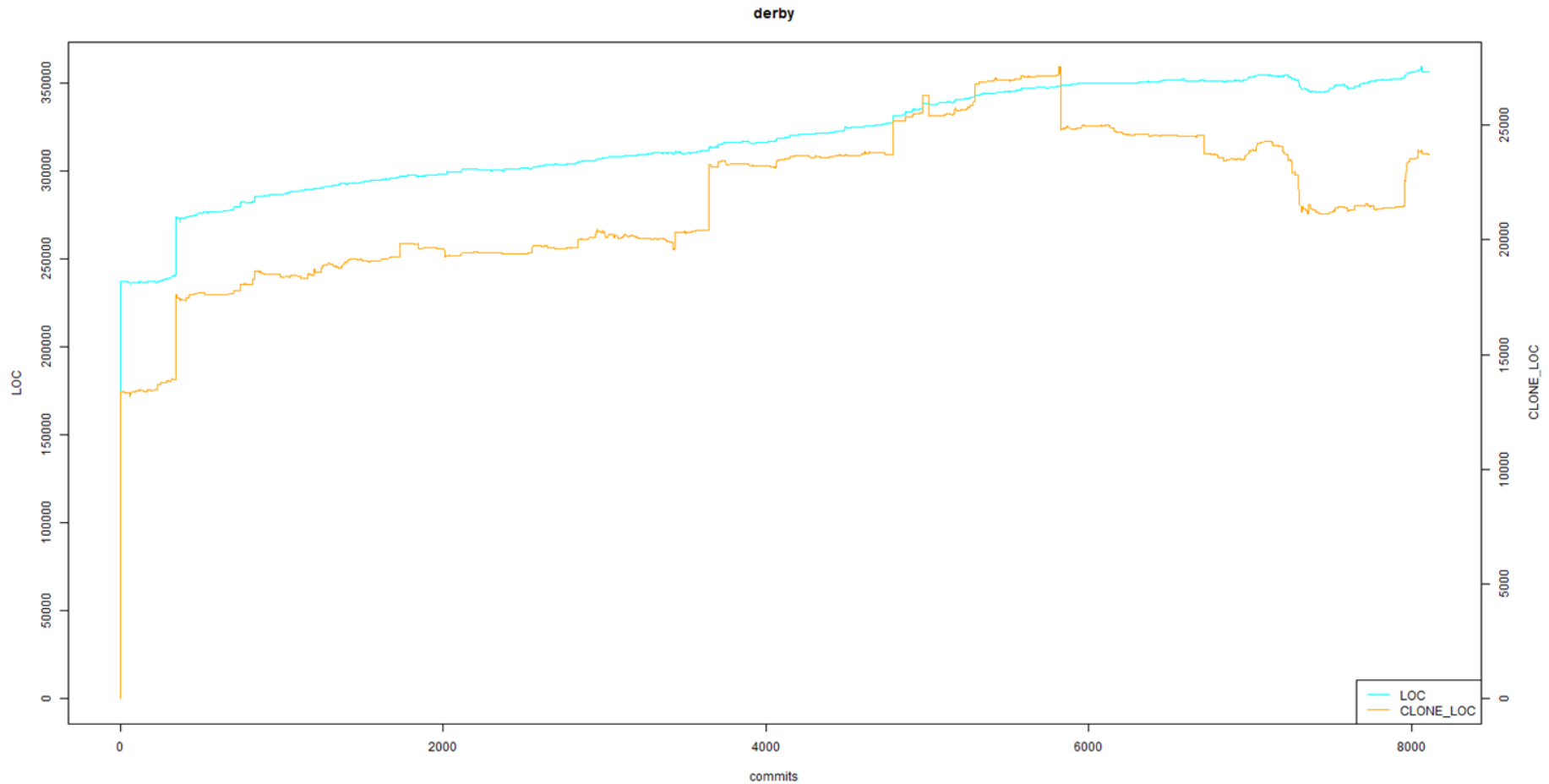


- (1) コード片を特定
- (2) クローンを特定
- (3) クローンを追跡
- (4) クローンの作成・利用に関与した開発者を特定
- (5) バグが混入したコミットを特定
- (6) バグと開発者とクローンの紐づけ

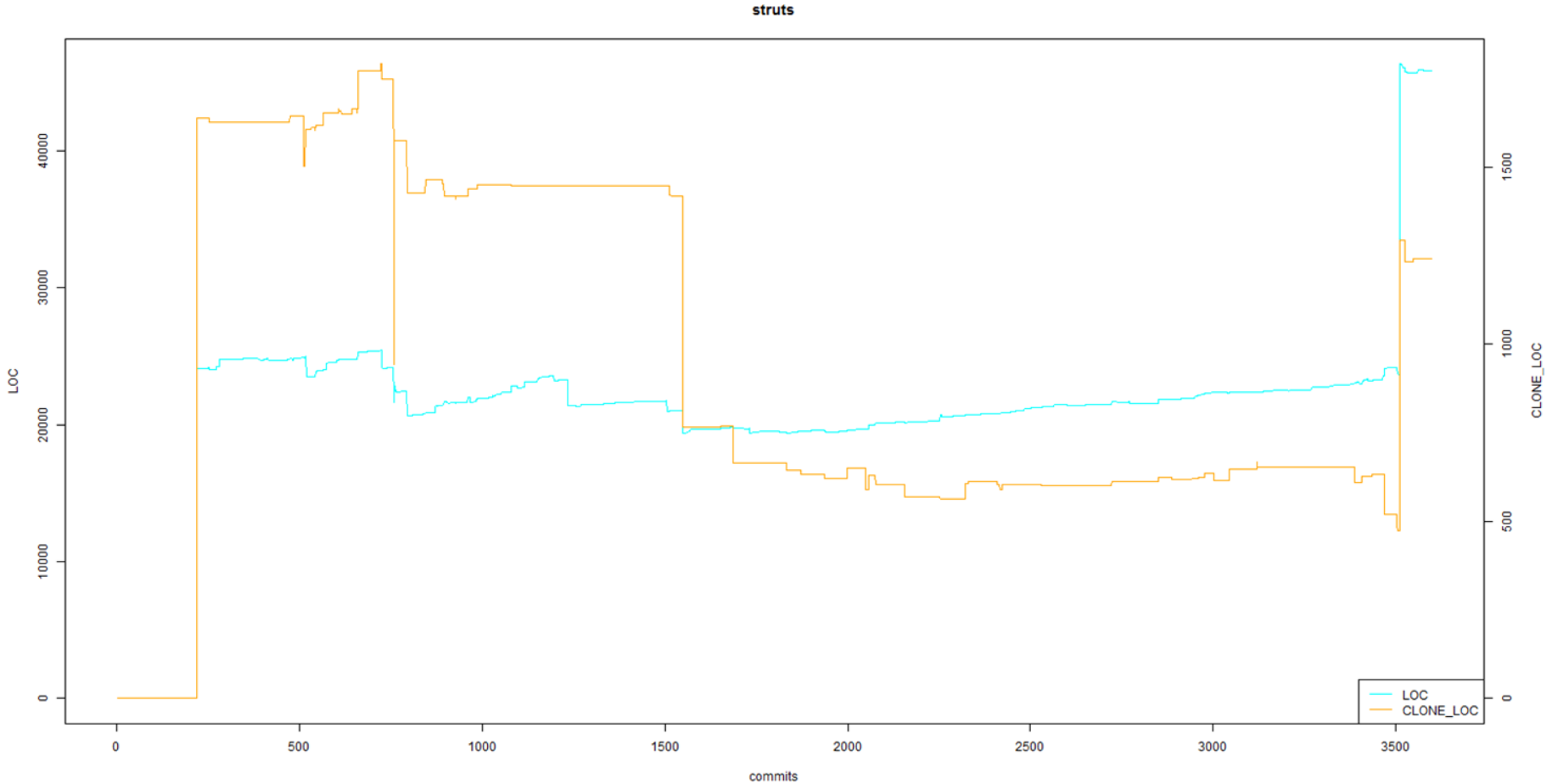
解析対象プロジェクト

プロジェクト名	期間	開発者数	ファイル数 (最新Rev)	クローン数 (最新Rev)	バグ有クローン数 ／ バグ無クローン数
Derby	2004/08 ~2016/06	37	1807	557	846/961
Struts	2006/02~ 2016/04	61	740	81	122/71

DerbyのLOCとCLONELOC



StrutsのLOCとCLONELOC



コードクローン追跡の例

- 修正漏れの例

created : Dag H. Wanvik

DERBY- 4741 (Bug) - reporter Dag H. Wanvik
: Make Derby work reliably in the presence of
thread interrupts

```
2075 } catch (InterruptedException e) {  
2076     // Retry. We needed to back all the way up here in the case  
2077     // of the (file) container having been closed due to an  
2078     // interrupt since the recovery needs the monitor to  
2079     // allocCache  
2080     if (--maxTries > 0) {  
2081         done = false;  
2082  
2083         // Wait a bit so recovery can take place before  
2084         // we re-grab monitor on "this" (which recovery  
2085         // needs) and retry writeRAFHeader.  
2086         try {  
2087             Thread.sleep(INTERRUPT_RETRY_SLEEP);  
2088         } catch (InterruptedException ee) {  
2089             // This thread received an interrupt as  
2090             // well, make a note.  
2091             InterruptStatus.setInterrupted();  
2092         }  
2093         continue;  
2094     } else {  
2095         throw StandardException.newException(  
2096             SQLState.FILE_IO_INTERRUPTED, e);  
2097     }  
2098 }  
2099 }
```

```
564 } catch (InterruptedException e) {  
565     if (--maxTries > 0) {  
566         success = false;  
567  
568         // Wait a bit so recovery can take place before  
569         // we re-grab monitor on "this" (which recovery  
570         // needs) and retry writeRAFHeader.  
571         try {  
572             Thread.sleep(500); // 0.5s  
573         } catch (InterruptedException ee) {  
574             // This thread received an interrupt as  
575             // well, make a note.  
576             InterruptStatus.setInterrupted();  
577         }  
578         continue; // retry write of RAFHeader  
579     } else {  
580         // We have tried for a minute, not sure what's  
581         // going on, so to be on safe side we can't  
582         // continue  
583         throw StandardException.newException(  
584             SQLState.FILE_IO_INTERRUPTED, e);  
585     }  
586 } catch (IOException ioe) {  
587 }
```

changed : Dag H. Wanvik

572	572		try {
573	-		Thread.sleep(500); // 0.5s
573	+		Thread.sleep(INTERRUPT_RETRY_SLEEP);
574	574		} catch (InterruptedException ee) {



コードクローン追跡の例

- 別のコードの修正によっておこる呼び出し方法の変更

DERBY-5498 (Bug) - reporter Knut Anders Hatlen
: ClosedByInterruptException in AuthenticationTest

co-changed: Dag H. Wanvik

```
1676 1674      try {  
1677      -         Thread.sleep(INTERRUPT_RETRY_SLEEP);  
1675      +         Thread.sleep(  
1676      +             InterruptStatus.INTERRUPT_RETRY_SLEEP);  
1678 1677      } catch (InterruptedException ee) {
```

```
514 514      try {  
515      -         Thread.sleep(INTERRUPT_RETRY_SLEEP);  
515      +         Thread.sleep(  
516      +             InterruptStatus.INTERRUPT_RETRY_SLEEP);  
516 517      } catch (InterruptedException ee) {
```

コードクローン追跡の例

- Initial Commitから変更がなく、バグ修正のためクローンのコード片が削除される

created : Dag H. Wanvik

java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java

```
538         if (init_startKeyValue == null) ↓
539         { ↓
540             // No start given, position at last slot + 1 of rightmost leaf ↓
541             pos.current_leaf = (LeafControlRow) root.searchRight(this); ↓
542         ↓
543             pos.current_slot = pos.current_leaf.page.recordCount(); ↓
544             exact = false; ↓
545         } ↓
```

java/engine/org/apache/derby/impl/store/access/btree/BTreeMaxScan.java

```
303         if (init_startKeyValue == null) ↓
304         { ↓
305             // No start given, position at last slot + 1 of rightmost leaf ↓
306             pos.current_leaf = (LeafControlRow) root.searchRight(this); ↓
307         ↓
308             pos.current_slot = pos.current_leaf.page.recordCount(); ↓
309             exact = false; ↓
310         } ↓
```



delete : Knut Anders Hatlen

```
519 -         if (init_startKeyValue == null)
520 -         {
521 -             // No start given, position at last slot + 1 of rightmost leaf
522 -             pos.current_leaf = (LeafControlRow) root.searchRight(this);
523 -
524 -             pos.current_slot = pos.current_leaf.page.recordCount();
525 -             exact = false;
526 -         }
```

DERBY-4193 reporter Kathey Marsden
: ASSERT FAILED Scan position already saved
with multi-threaded insert/update/delete