XII международная конференция

CEE-SECR / РАЗРАБОТКА ПО

28 - 29 октября, Москва

Устойчивая привязка к синтаксическим конструкциям в изменяющемся коде

Михаил Малеванный

Академия Строительства и Архитектуры

Донской Государственный Технический Университет

Введение

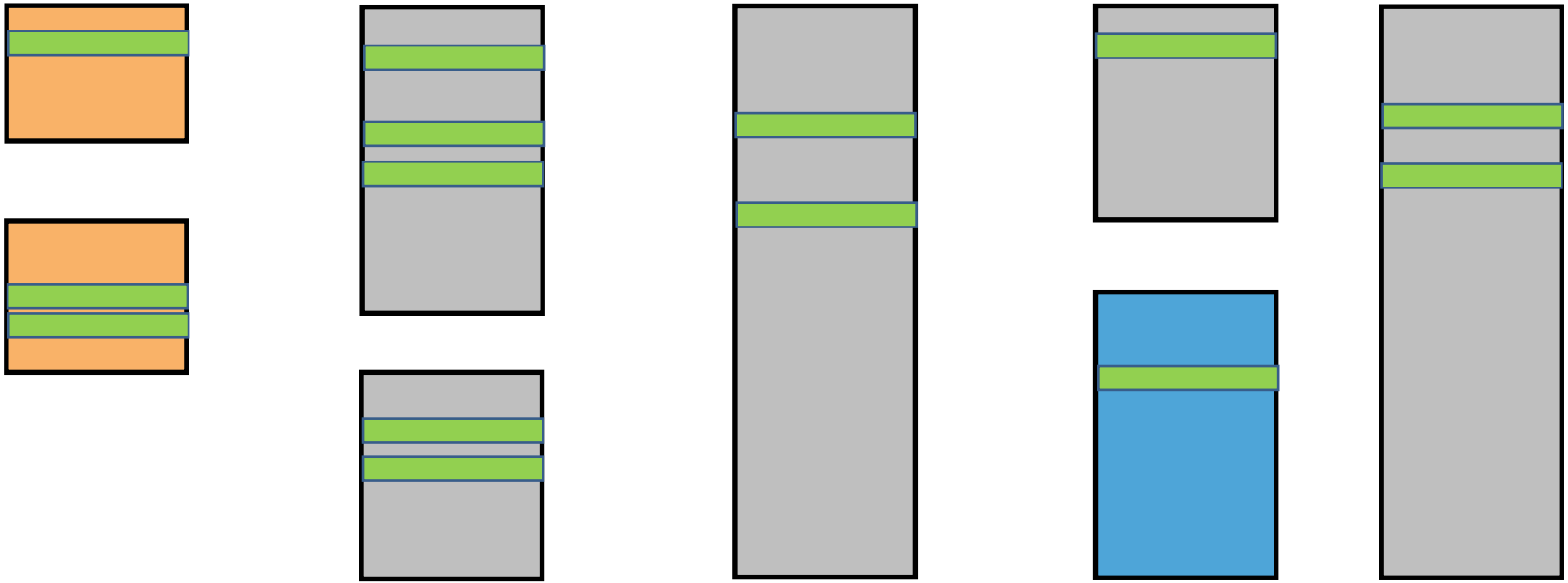
Предыдущий доклад (SECR-2015):

«Аспектная разметка кода для быстрой навигации по проекту»

Реализация:

https://github.com/MikhailoMMX/AspectMarkup

Сквозная функциональность



Рабочее множество

Рабочее множество...

...его:

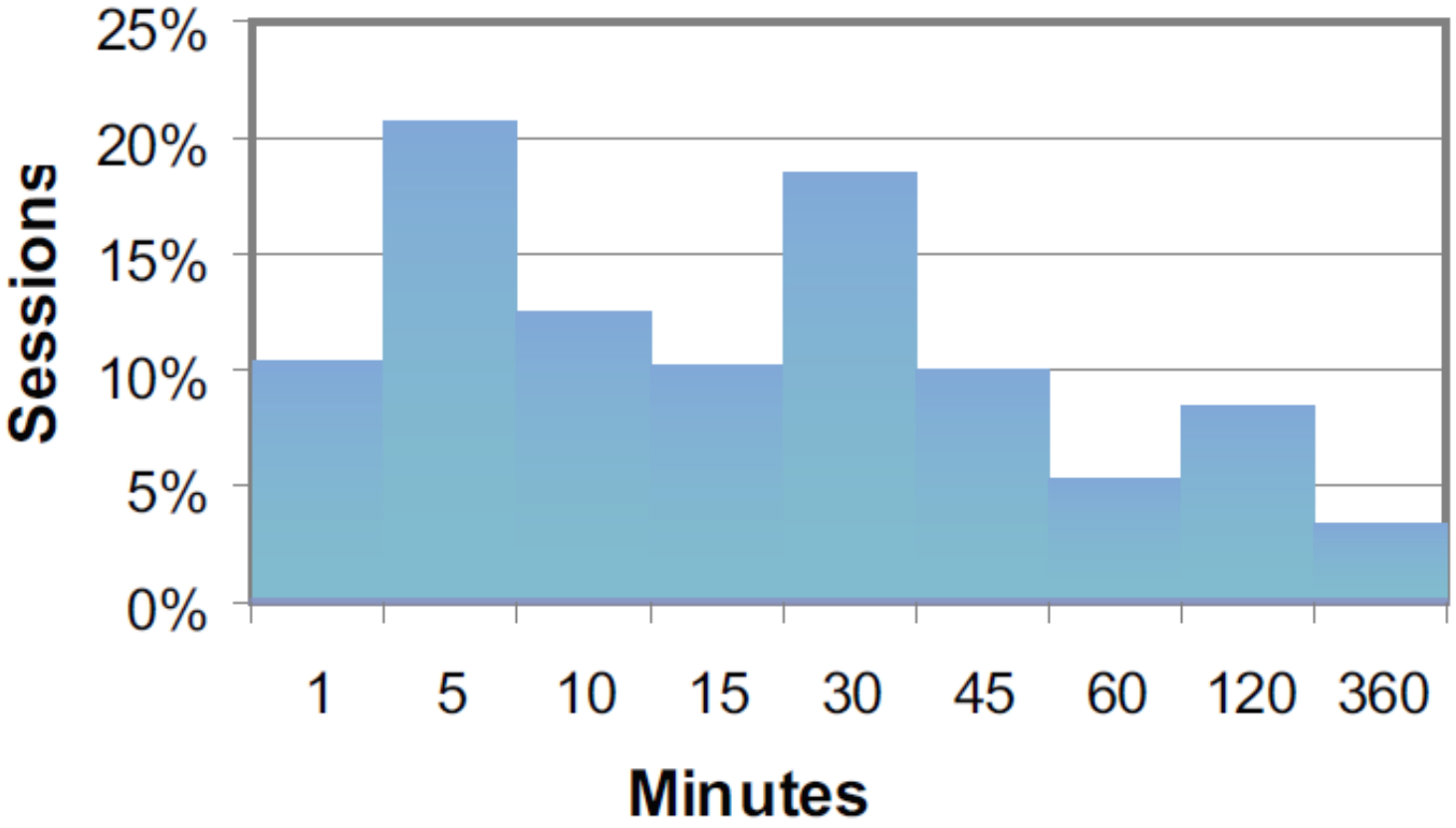
- Трудно найти
- Легко потерять
- Невозможно ~~забыть~~ вспомнить

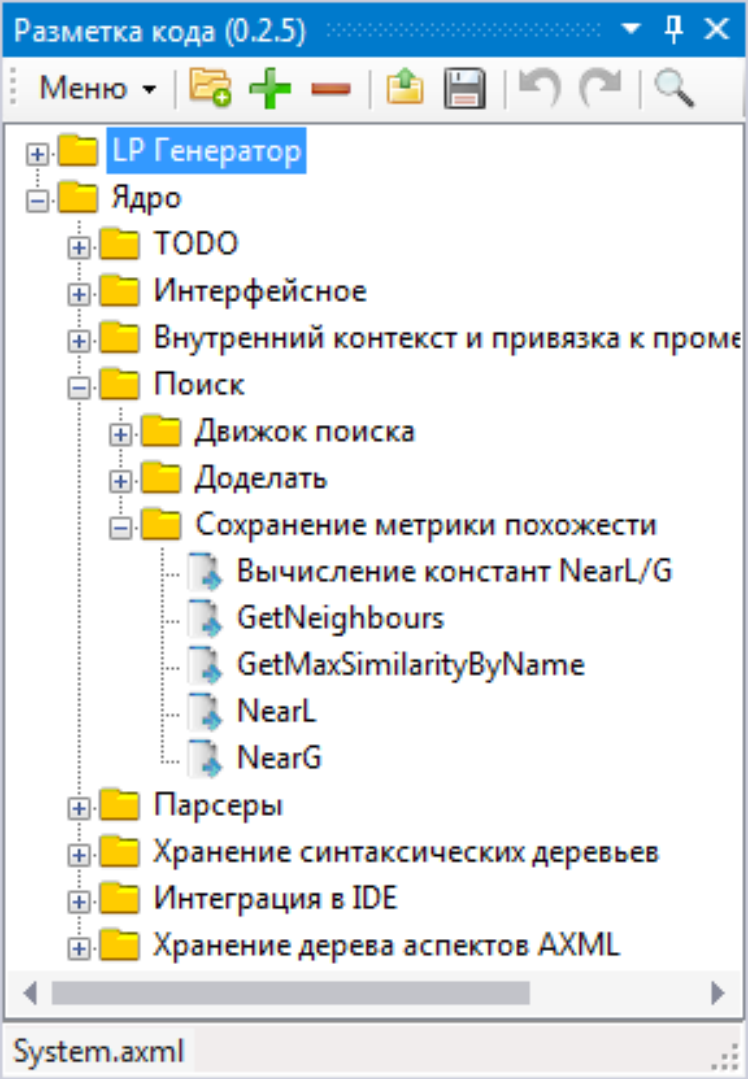
Chris Parnin and Spencer Rugaber. **Resumption strategies for interrupted programming tasks**. *Software Quality Journal* 19, 1 (March 2011), 5-34. DOI=http://dx.doi.org/10.1007/s11219-010-9104-9

«Только в 7% случаев перед редактированием **не** выполняется навигация по коду»

«Только в 10% случаев активная деятельность начинается в пределах минуты после возврата к задаче»

Edit Lag Distribution





Разметка кода

560 полей и методов
206 методов **visit**

visit(assign_tuple asstup)
visit(array_type _array_type)
visit(assign _assign)
visit(assign_tuple asstup)
visit(assign_var_tuple asstup)
visit(attribute _attribute)
visit(attribute_list _attribute_list)
visit(bin_expr _bin_expr)
visit(block _block)
visit(bool_const _bool_const)
visit(bracket_expr _bracket_expr)
visit(c_for_cycle node)
visit(c_module node)
visit(c_scalar_type node)
visit(case_node _case_node)
visit(case_variant _case_variant)
visit(case_variants _case_variants)
visit(char_const _char_const)

Разметка кода (0.2.5)

Меню

- !Исправленные ошибки
- Добавляем t[1] где t - Tuple
- Реализуем (a,b) := (1,2)
  - visit(assign_tuple
  - StartsWith("System.Tuple"
- Реализуем тип (integer,integer)
- Делаем ?System ищется только в .NET
- foreach
- Result :=
- QuestionPoint - незаконченное
- Выбор перегруженного метода
- Короткое определение функции
- x->Print(x)
- Преобразование синт дерева
- Yield
- a.Sort((row1, row2) -> 1)
- foreach Arr -> for
- [::]
- Таблица символов
- Ядро преобраз-я синт. деревьев
  - visit(SyntaxTree.dot_node
  - visit(SyntaxTree.indexer
  - // Замена foreach на for для массива
  - visit(vdarr);
  - foreach (statement syntax_statement

PascalABCNET.axml

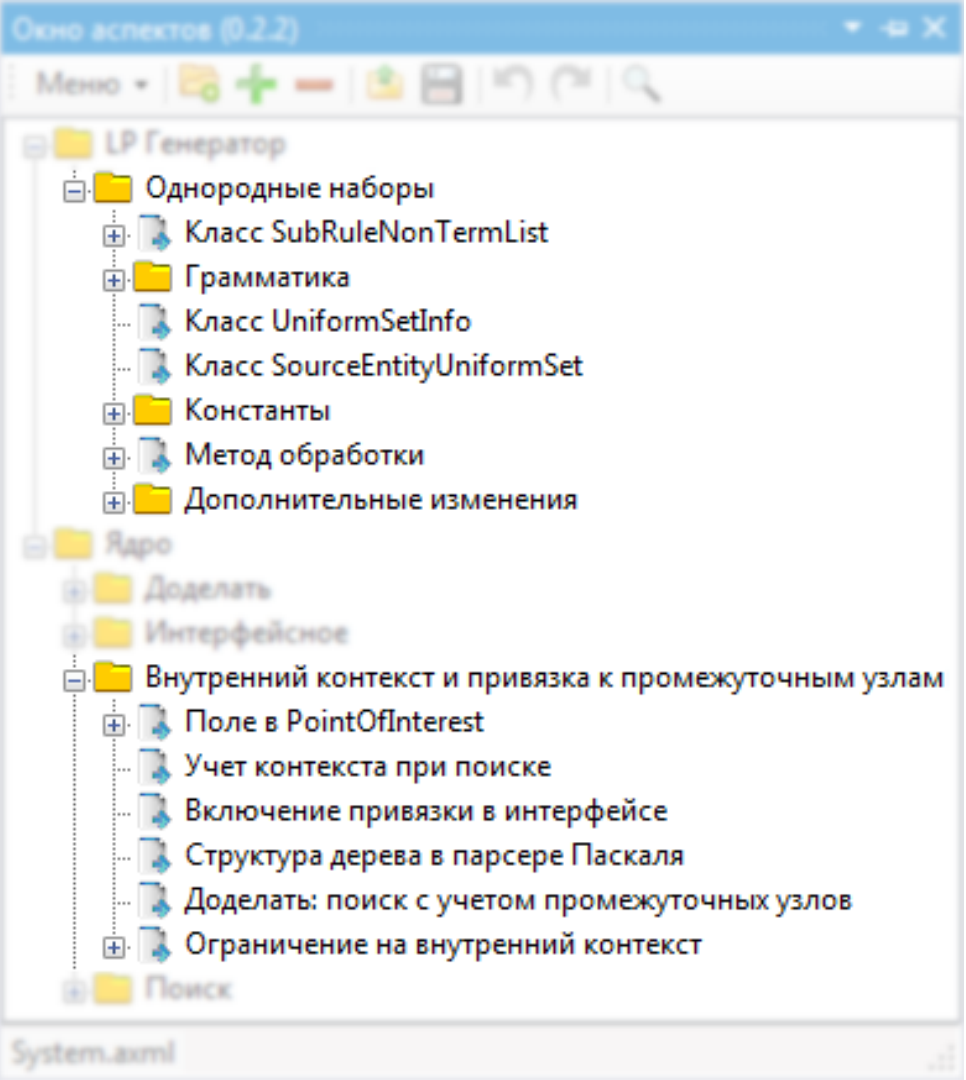
ProjectInfo.cs | syntax_tree_visitor.cs | CompilerInternalErrorReport.resx | Interfaces.cs

TreeConverter | PascalABCCompiler.TreeConverte | visit(assign_tuple asstup)

Реализуем (a,b) := (1,2)
visit(assign_tuple
StartsW ("System.Tuple"

```
public override void visit(SyntaxTree.assign_tuple asstup)
{
    // Проверить, что справа - Tuple
    var expr = convert_strong(asstup.expr);
    expr = convert_if_typed_expression_to_function_call(expr);
    var t = ConvertSemanticTypeNodeToNETType(expr.type);
    if (t == null)
```

100 %



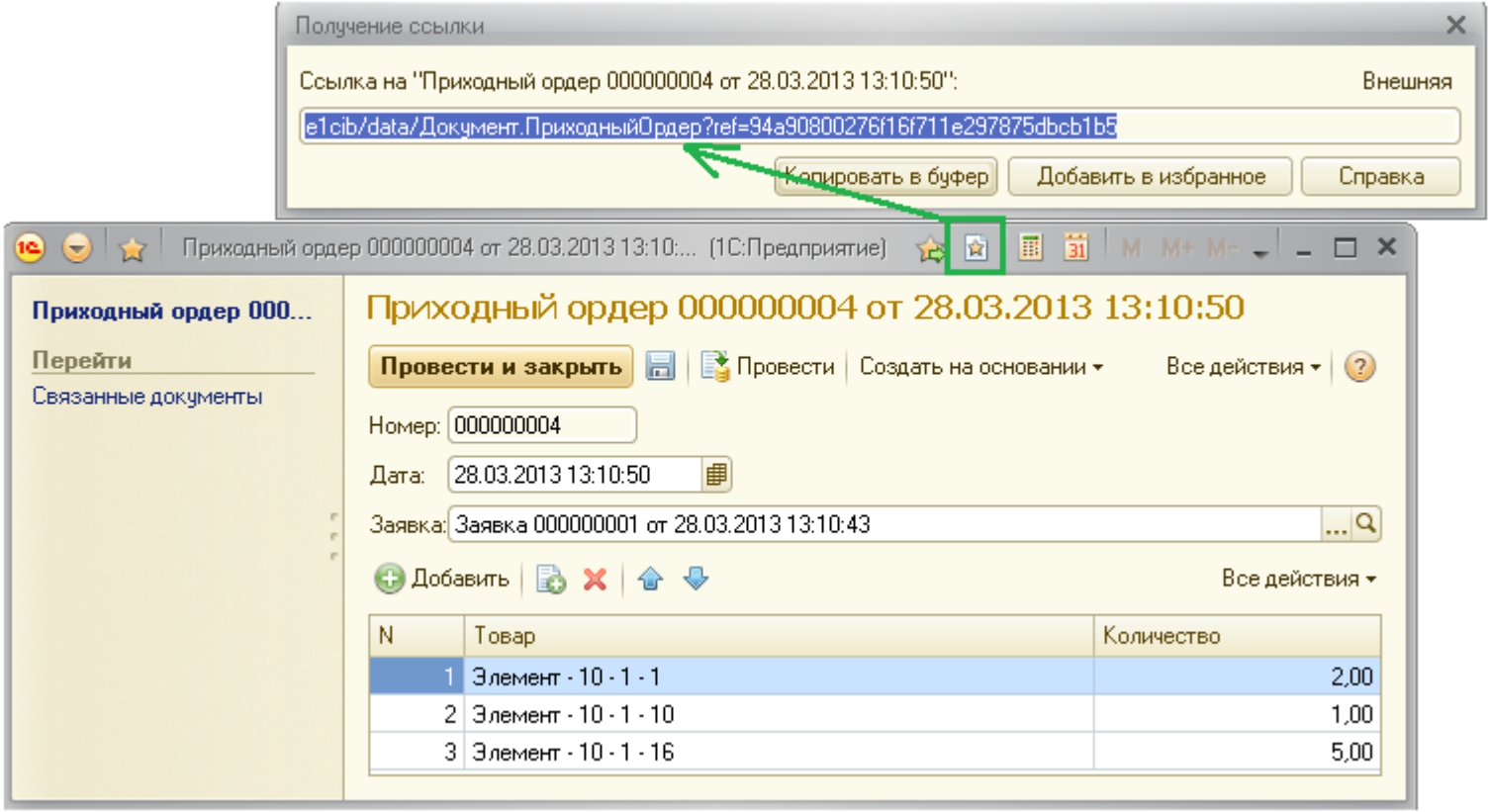
Задача 1

Рабочее множество 1

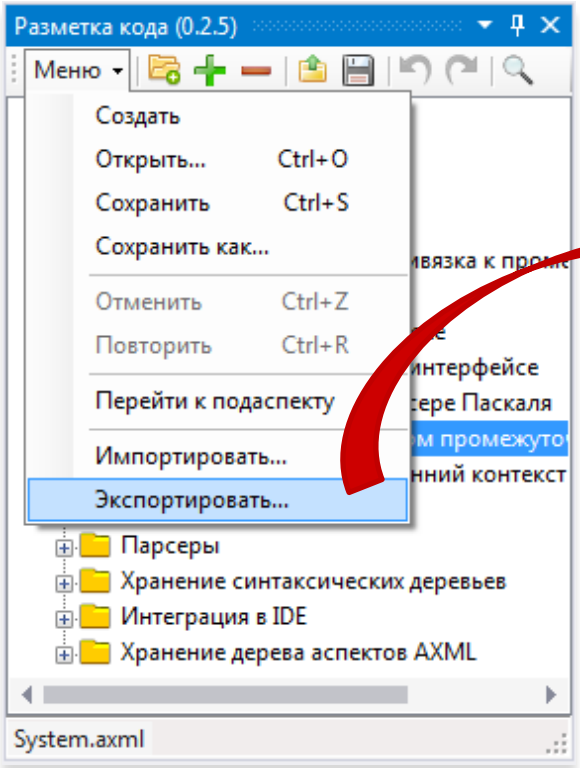
Задача 2

Рабочее множество 2

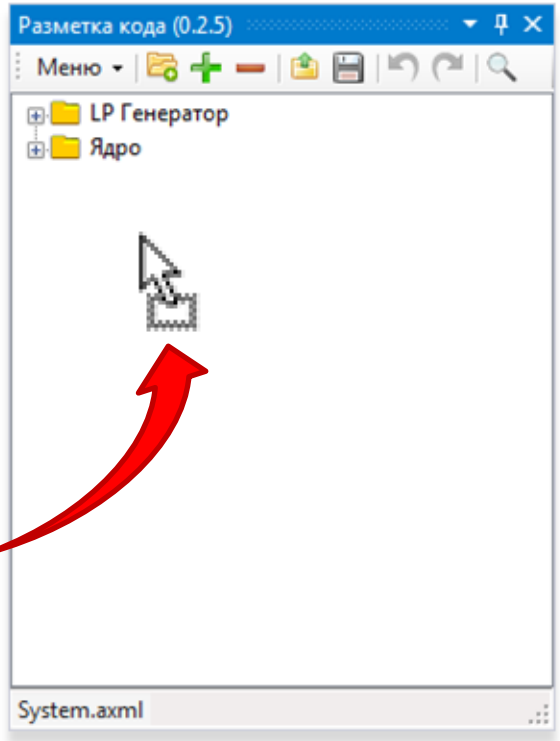
Навигационные ссылки в 1С:Предприятие



Навигационная ссылка в IDE



- Почта
- Баг-трекер
- …

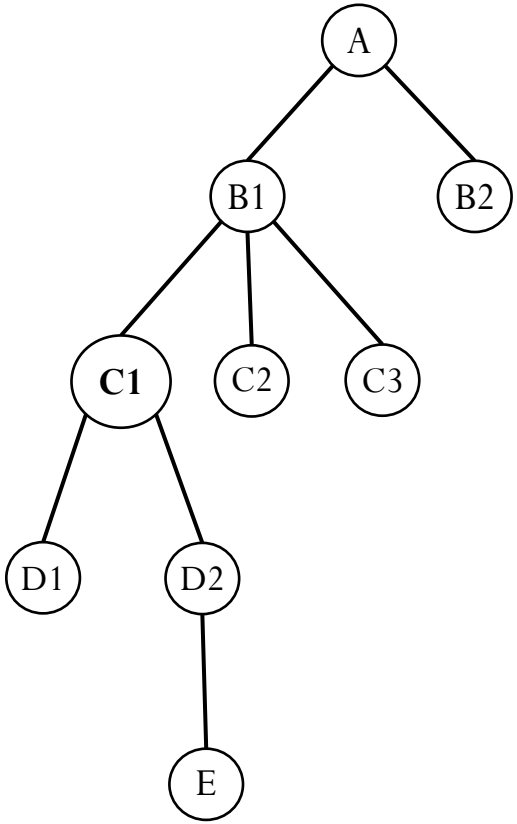


Модель привязки

- Сохраняемая информация

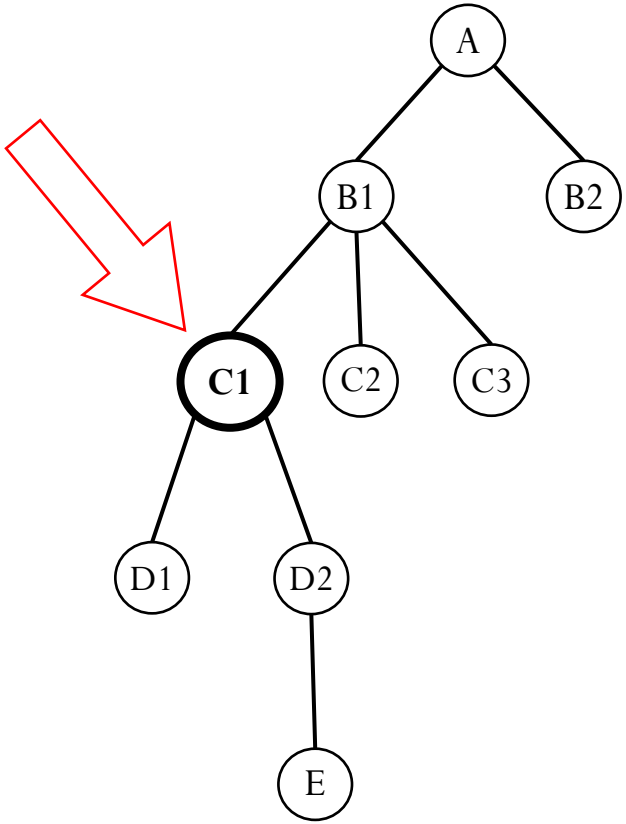
Синтаксическое дерево

- Строится легковесным парсером по исходному коду
- Общее для разных языков
- Не зависит от среды разработки



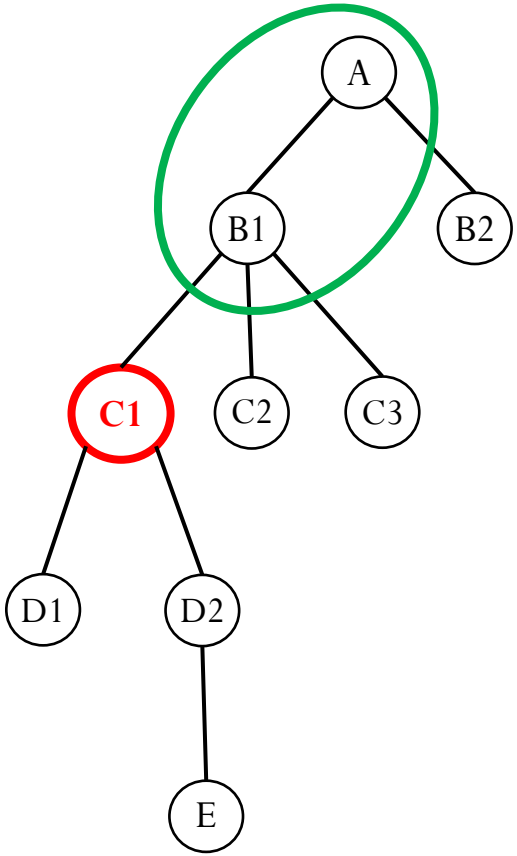
Сохраняемая информация

- Имя + заголовок
- Тип
- Контекст
  - Родительские узлы
  - Соседние узлы
  - Дочерние узлы



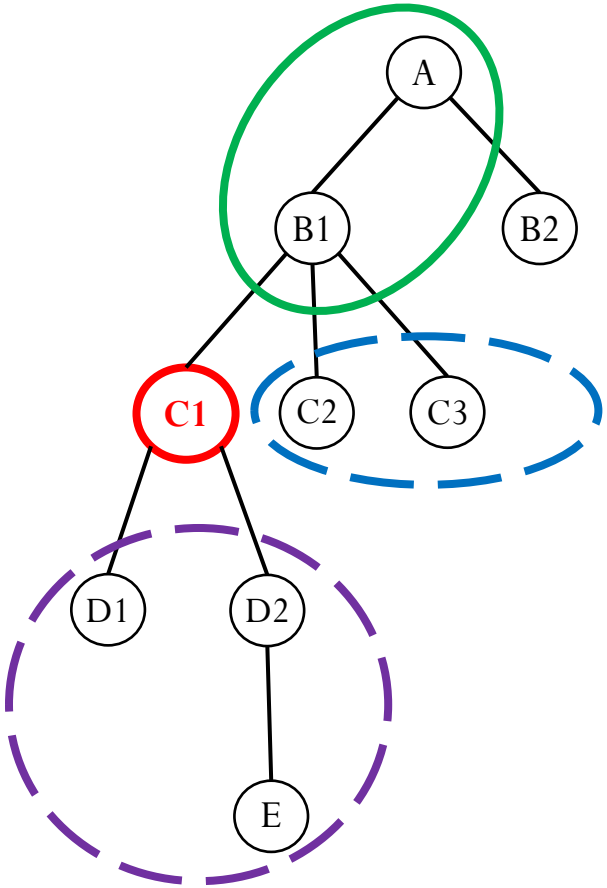
Сохраняемая информация

- **Имя + заголовок**
- **Тип**
- Контекст:
  - Родительские узлы



Сохраняемая информация

- **Имя + заголовок**
- **Тип**
- Контекст:
  - Родительские узлы
  - Соседние узлы *
  - Подузлы *



Имя и заголовок (*Java*)

Заголовок

```
public Component addChild(Component child)
{
   repository.connectComponents(this, child,
     EdgeKind.CONTAINS);
   return child;
}
private int id;
```

Имя

Типы узлов (*Java*)

Тип: **Method**

```
public Component addChild(Component child)
{
   repository.connectComponents(this, child,
     EdgeKind.CONTAINS);
   return child;
}
private int id;
```

Тип: **Field**

Внешний контекст (*C#*)

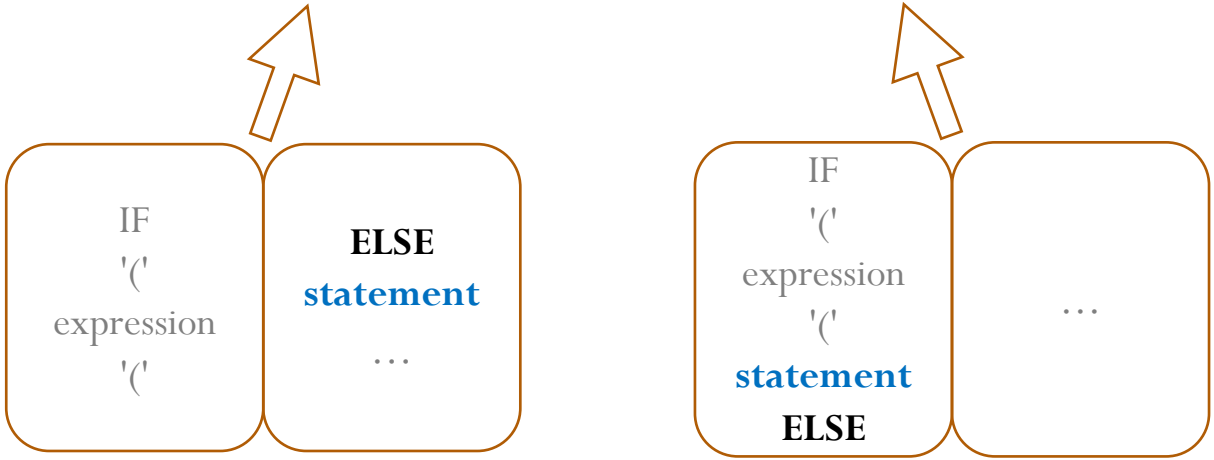
```
namespace N
{

    class C1 : IVisitor
    {
        public void visit(IVisitor v) { }
    }


    class C2 : IVisitor
    {
        public void visit(IVisitor v) { }
    }
}
```

- namespace N
- class **C1** : IVisitor

- namespace N
- class **C2** : IVisitor

Горизонтальный контекст (*yacc*)

```
selection_statement
   : IF '(' expression ')'
     statement ELSE statement
```



Внутренний контекст (*Pascal*)

- X,Y : Double

```
var
    X, Y : Double;
```

- Name, Address: String
- Age : Integer

```
var
    Name, Address : string;
    Age : integer;
```

Модель привязки

- Алгоритм поиска в изменяющемся коде

Величина похожести

T – исходный узел

$\mathbf{T_i}$ – узлы в новом файле

Для всех $\mathbf{T_i}$ вычисляем величину «похожести»:

$$\mathbf{S_i = Similarity(T, T_i)}$$

$\mathbf{S_i} \in [0, 1]$; 0 = ничего общего, 1 = точное совпадение

Редакционное расстояние

Минимальное количество правок, необходимых для преобразования одной строки в другую

Выше *Редакционное расстояние* – ниже *Похожесть*

3 версии:

- Имена (String)
- Заголовки (Lists<String>)
- Внешние контексты (List<Headers>)

Примеры метрики «*похожести*»

Строка 1	Строка 2	Похожесть
`String`	`String`	1
`load_file`	`loadFile`	0.77
`nextInt`	`nextDouble`	0.4
`String`	`Object`	0

Примеры метрики «*похожести*»

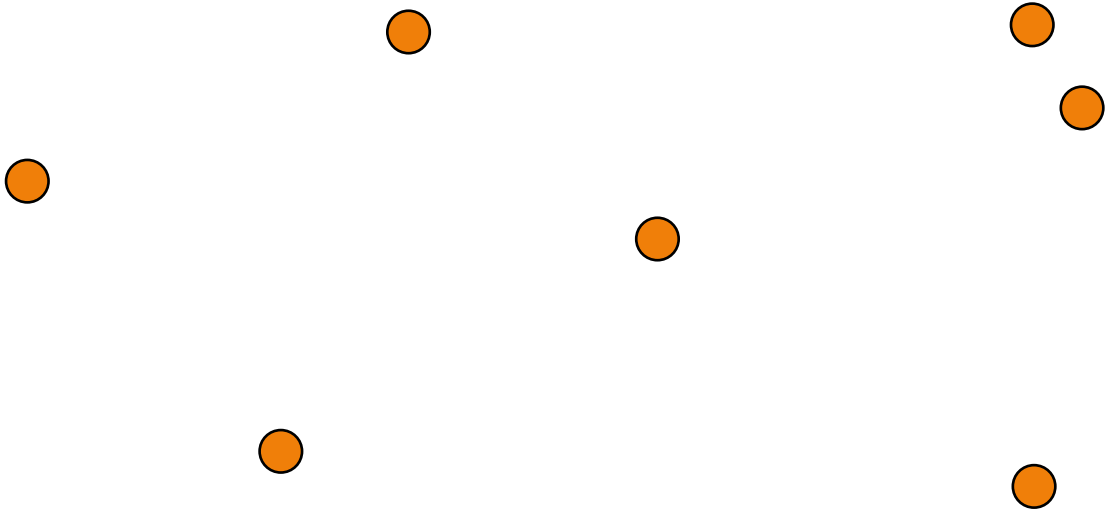
- private static uint GetManagedResourceOffset(ManagedResource resource, Blob**Writer** resourceWriter)
- private static uint GetManagedResourceOffset(ManagedResource resource, Blob**Builder** resourceWriter)

= 0.95

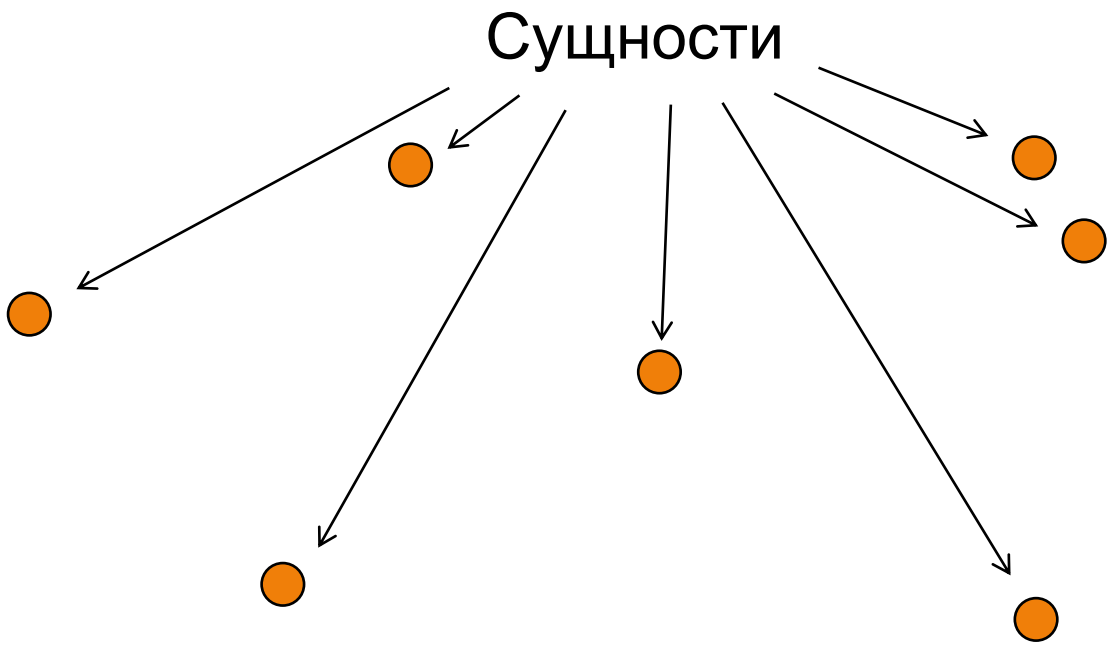
- public string GetLine(int LineIndex)
- public bool IsDocumentOpen(string FileName)

= 0.33

Аналогия с помехоустойчивыми кодами



Аналогия с помехоустойчивыми кодами



Аналогия с помехоустойчивыми кодами

int Count()

void visit(statement)

void visit(expression)

event EventHandler(…)

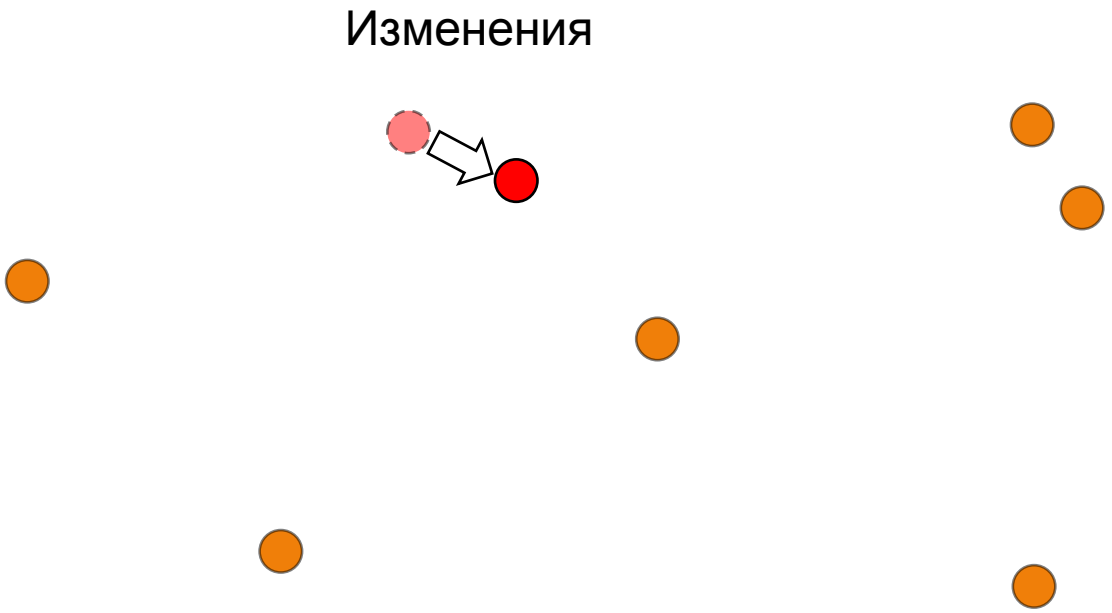
int GetHashCode()

async Task TestElseIf()

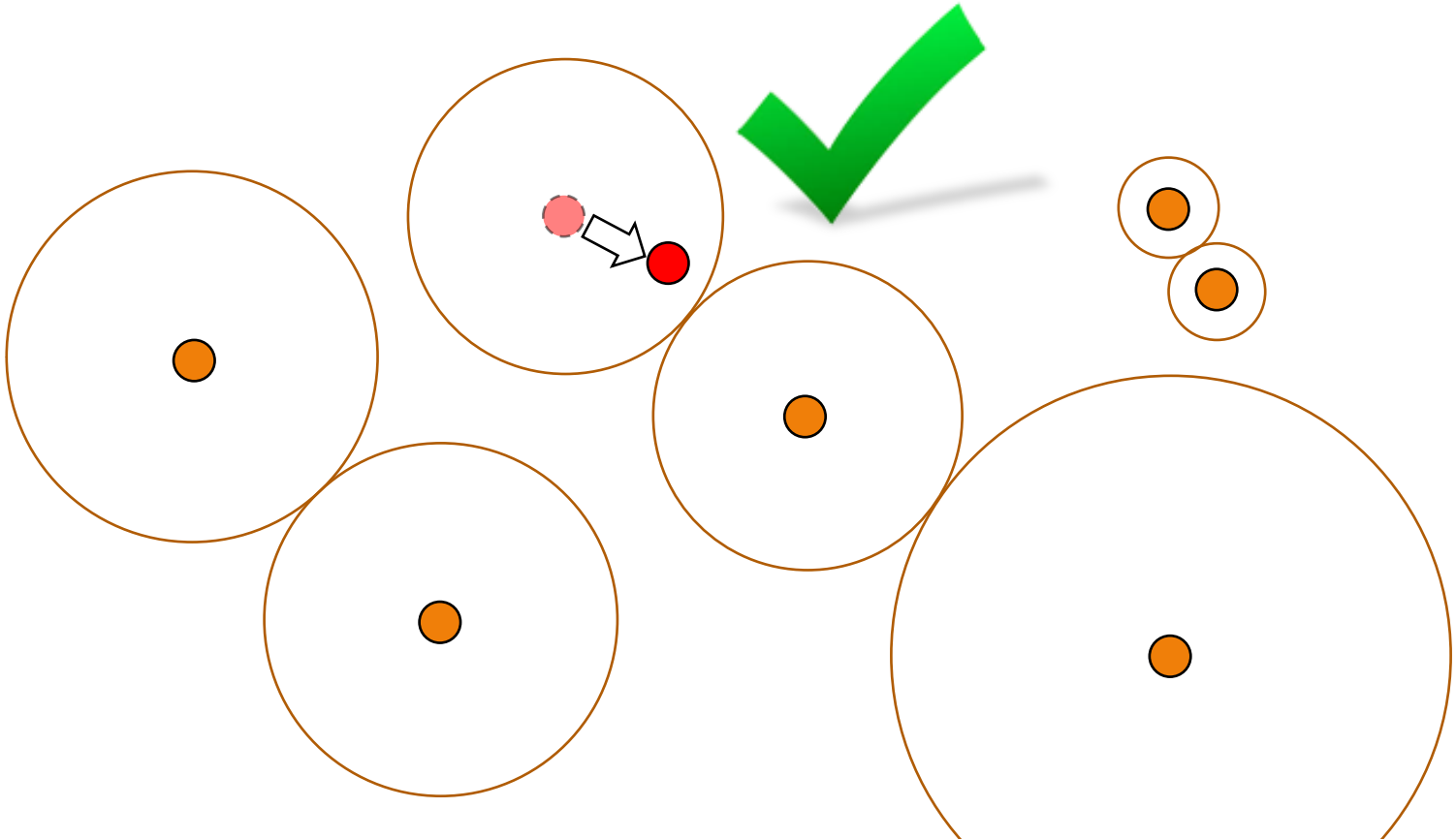
void WithAttributes()

Аналогия с помехоустойчивыми кодами

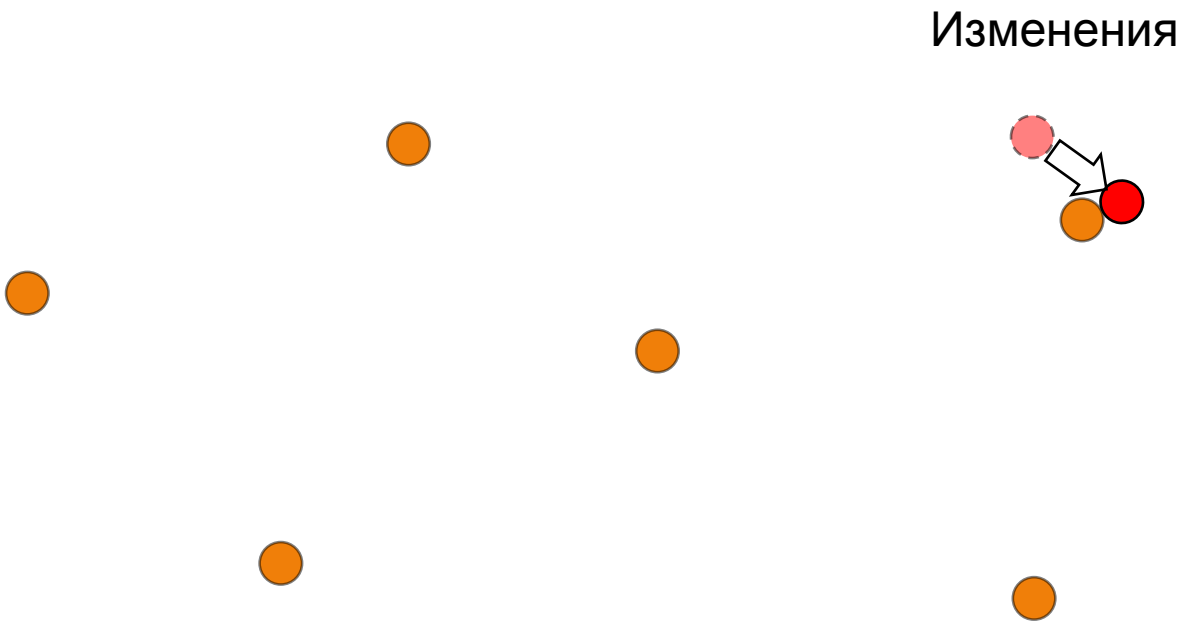
Аналогия с помехоустойчивыми кодами



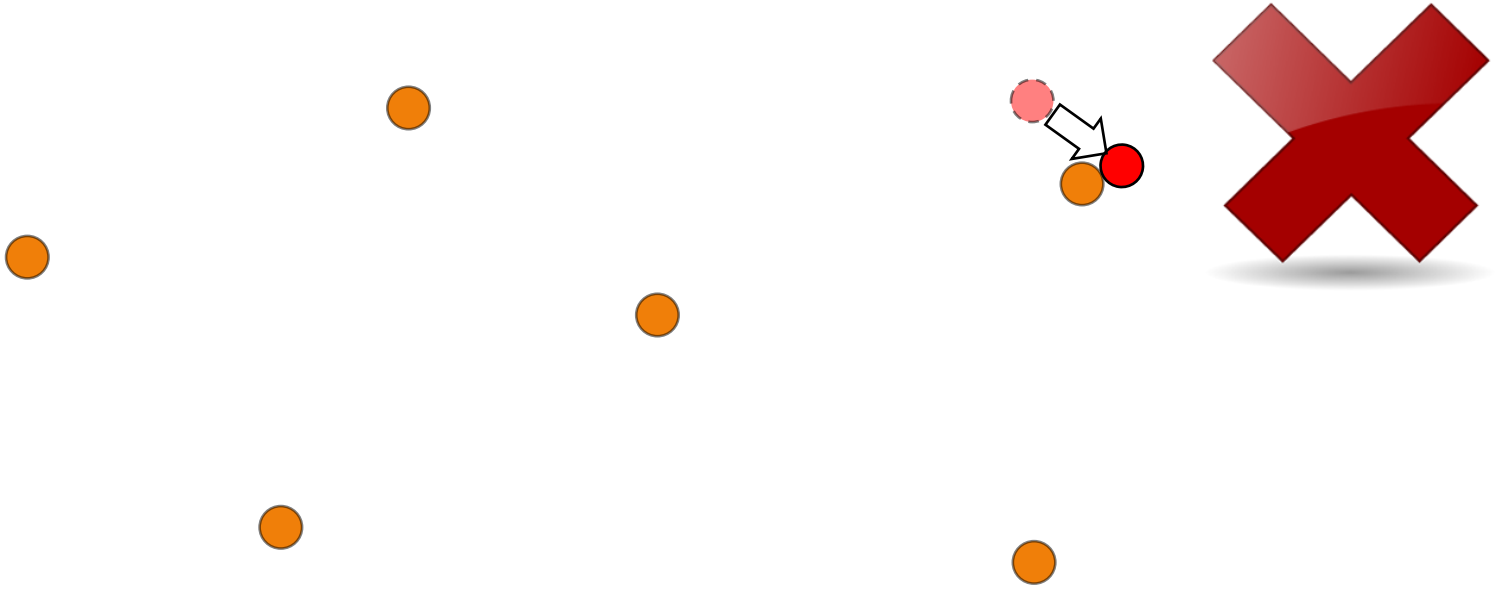
Аналогия с помехоустойчивыми кодами



Аналогия с помехоустойчивыми кодами



Аналогия с помехоустойчивыми кодами



Пример неоднозначности поиска

Первая версия:

```
class expr : Node
{
    public void Visit(Visitor _vis)
    { /* */ }
}
```

Переименован

Добавлен

Исходный фрагмент

Вторая версия:

```
class Expression : Node
{
    public void Visit(Visitor _vis)
    { /* */ }
}

class Statement : Node
{
    public void Visit(Visitor _vis)
    { /* */ }
}
```

Разрешение неоднозначности поиска

```
  %   Имя узла
100   public void Visit ( Visitor _vis )
 99   public void Visit ( Visitor _vis )
 98   public void Visit ( Visitor _vis )
 41   public override bool ReturnsVoid
 39   internal sealed override bool IsMetac
 38   public override bool IsOverride
 37   public override MethodKind MethodKinc
```

Разрешение неоднозначности поиска

```
  %   Имя узла
100   public void Visit ( Visitor _vis )
 99   public void Visit ( Visitor _vis )
 98   public void Visit ( Visitor _vis )
 41   public override bool ReturnsVoid
 39   internal sealed override bo
 38   public override
 37   public override MethodKind
```

Исходный узел из **_expr**

из **Expression**

из **Statement**

Исследование устойчивости привязки

Методика

Ревизии:

Initial commit

Ревизия 1 (начало анализа)

Исследуемый период

Ревизия 2 (конец анализа)

Привязка ко **всем** сущностям в ревизии 1

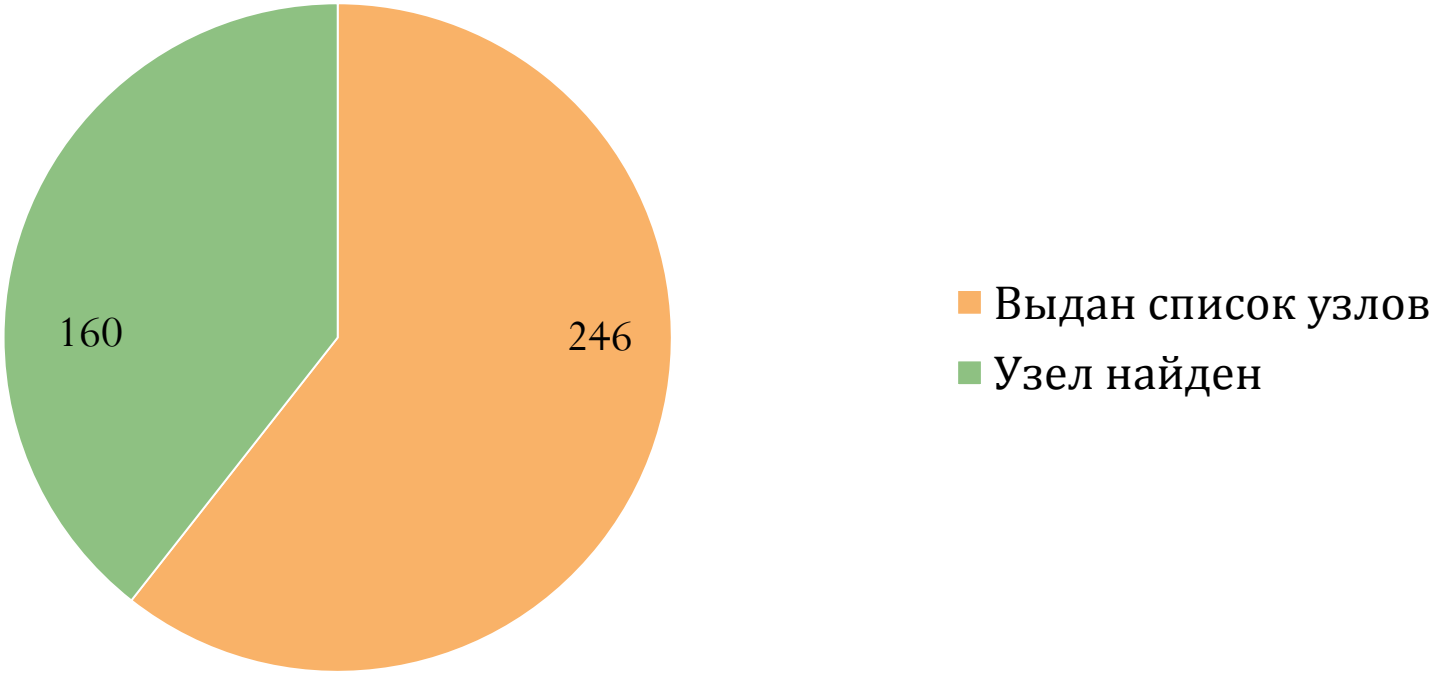
Проверка привязки в ревизии 2

Ревизии:

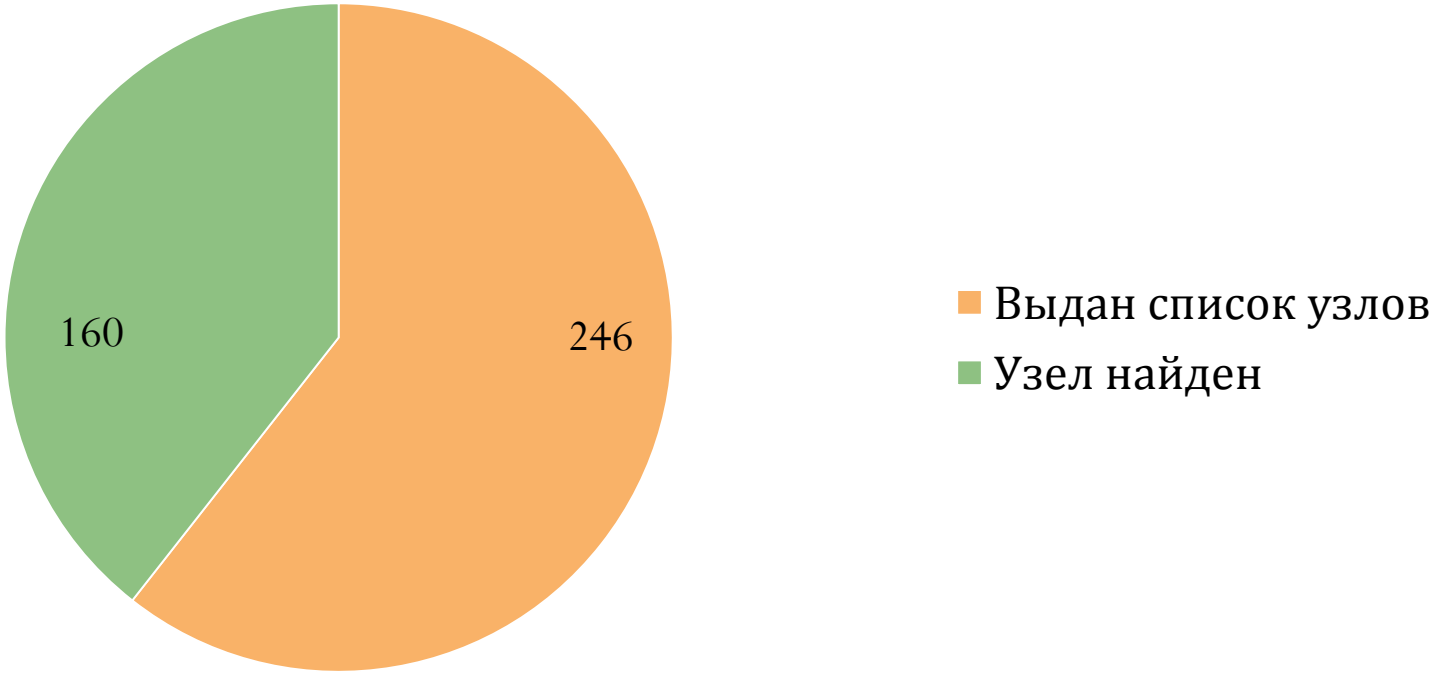
#1 Initial commit	#326 Начало анализа	9 месяцев разработки	#653 Конец анализа

- Изменено файлов **95** из 2 668
- Изменено сущностей **406** из 83 082 (0,5%)

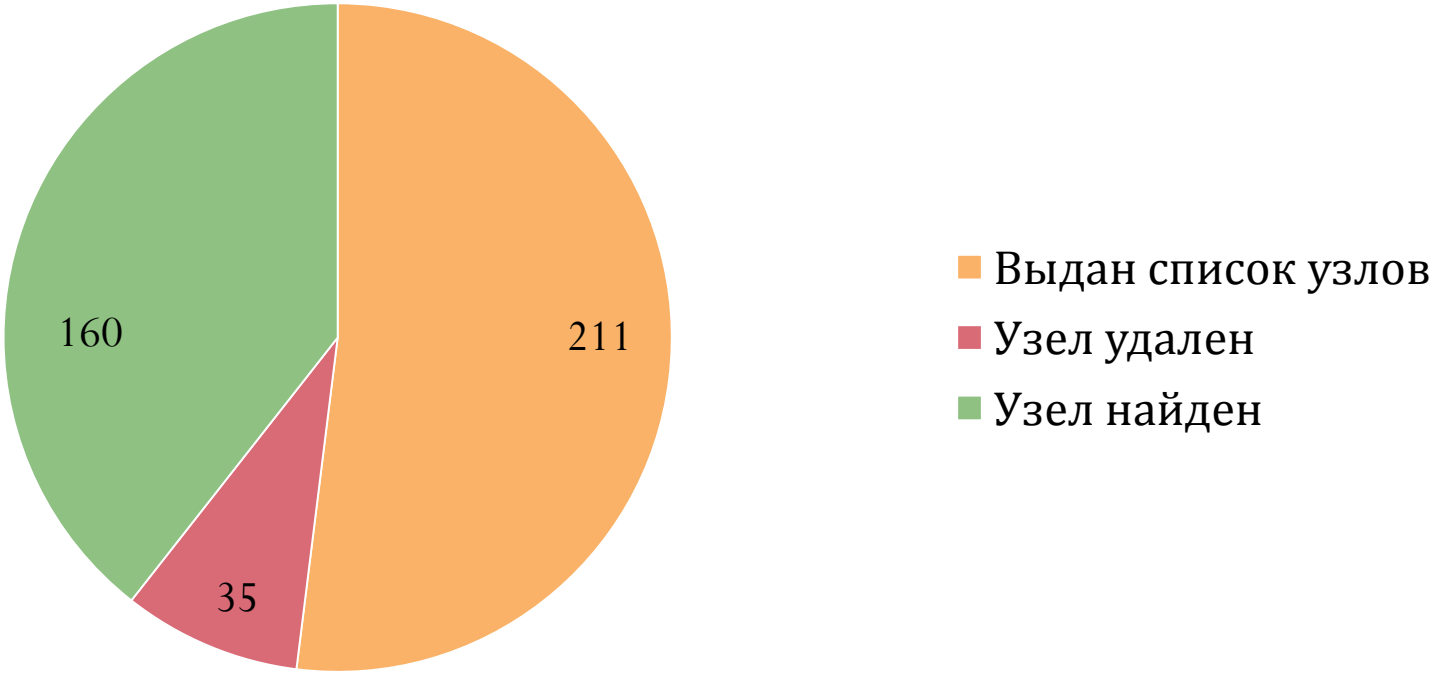
Предварительный результат поиска узлов



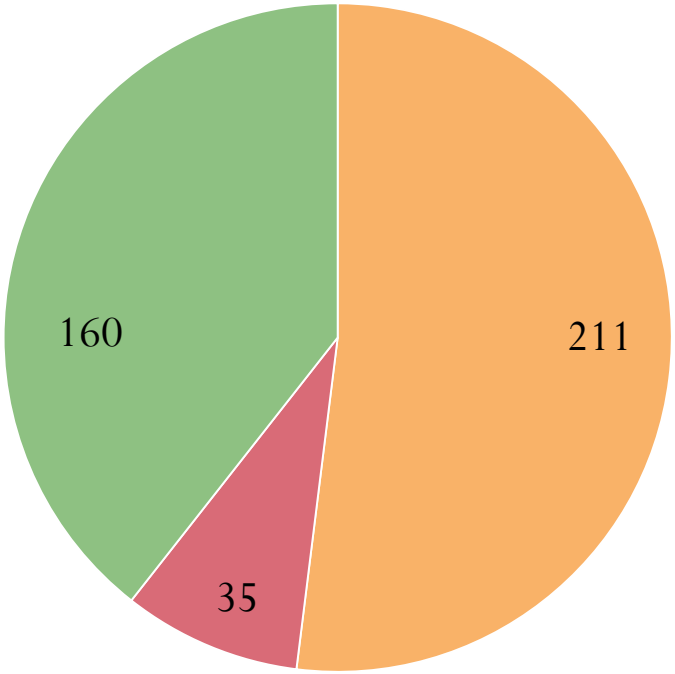
Предварительный результат поиска узлов



Удаленные узлы

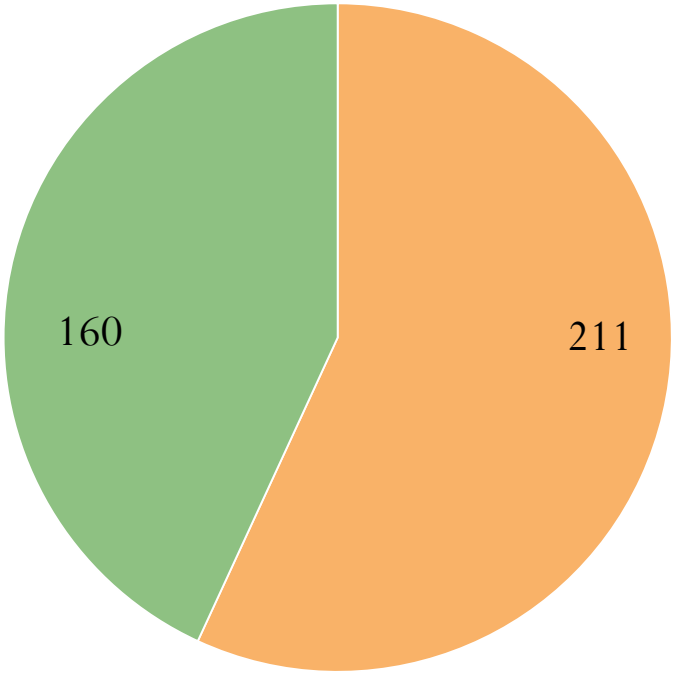


Удаленные узлы



- Выдан список узлов
- Узел удален
- Узел найден

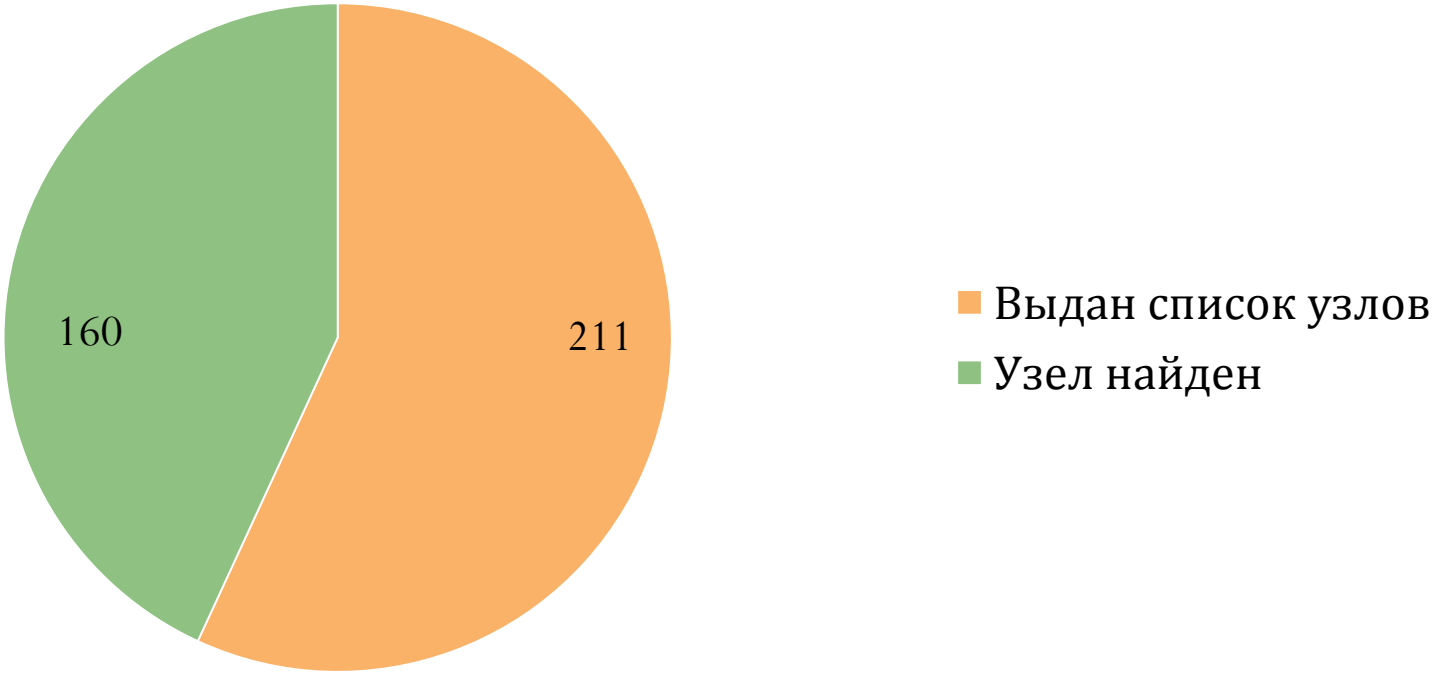
Без удаленных узлов



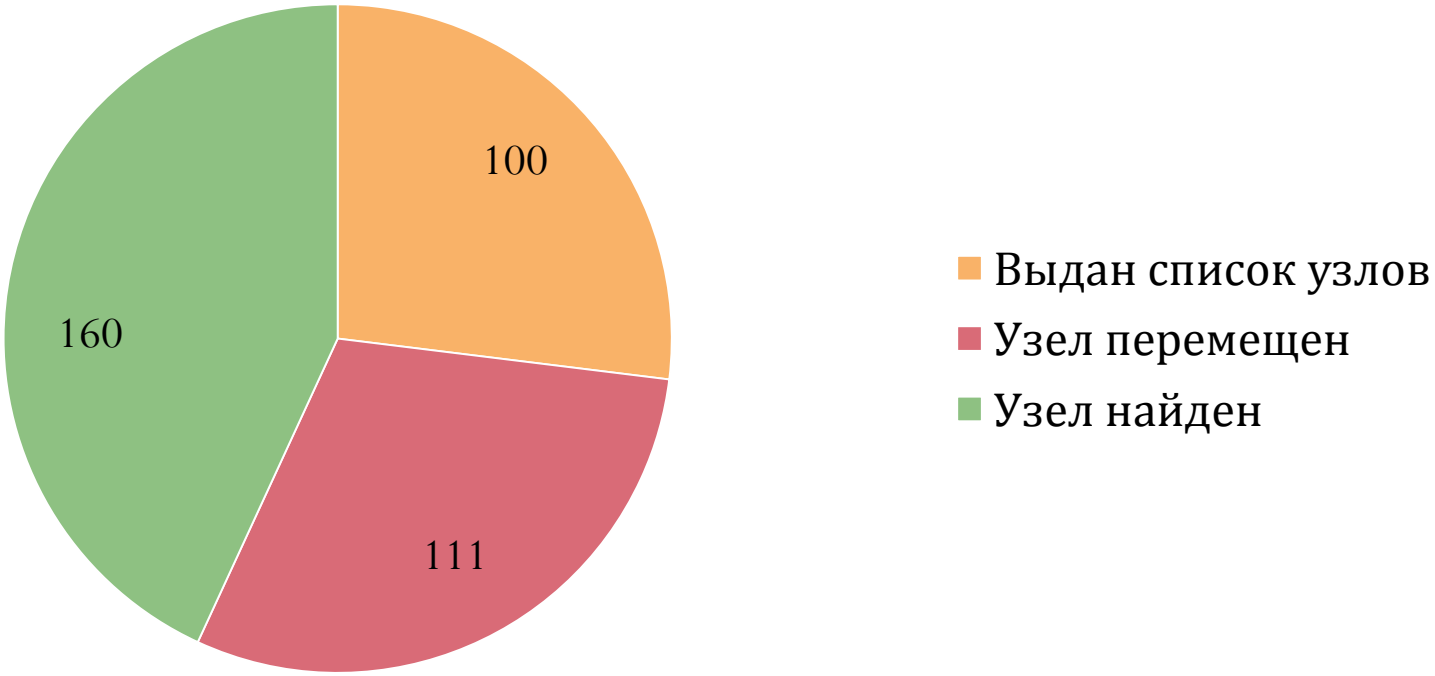
Выдан список узлов

Узел найден

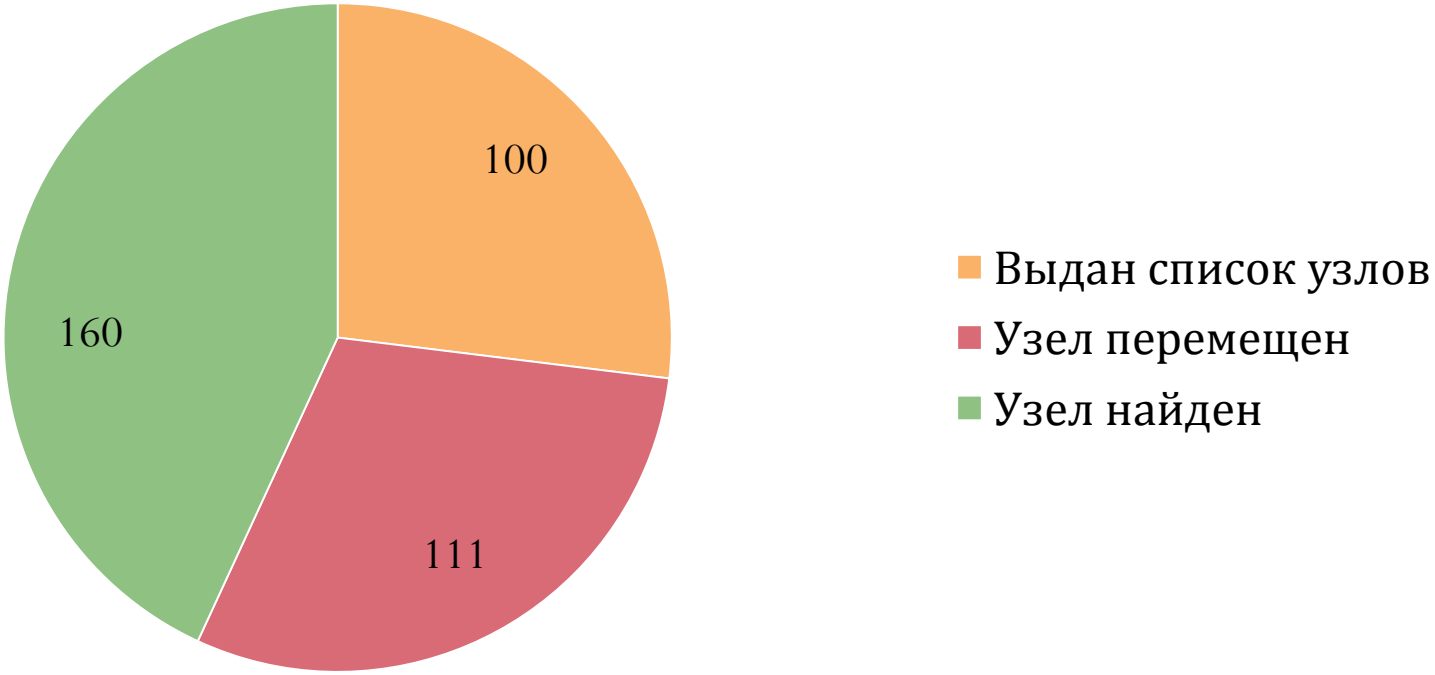
Без удаленных узлов



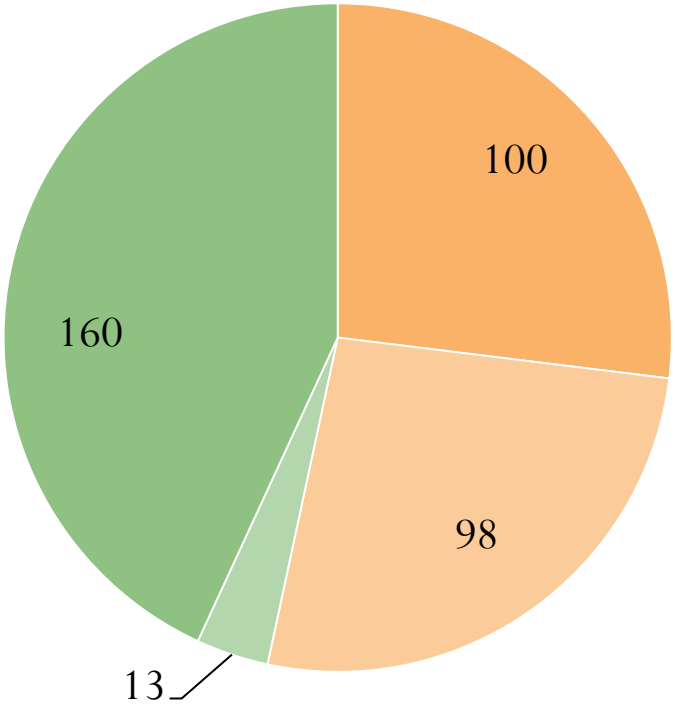
Перемещенные узлы



Перемещенные узлы

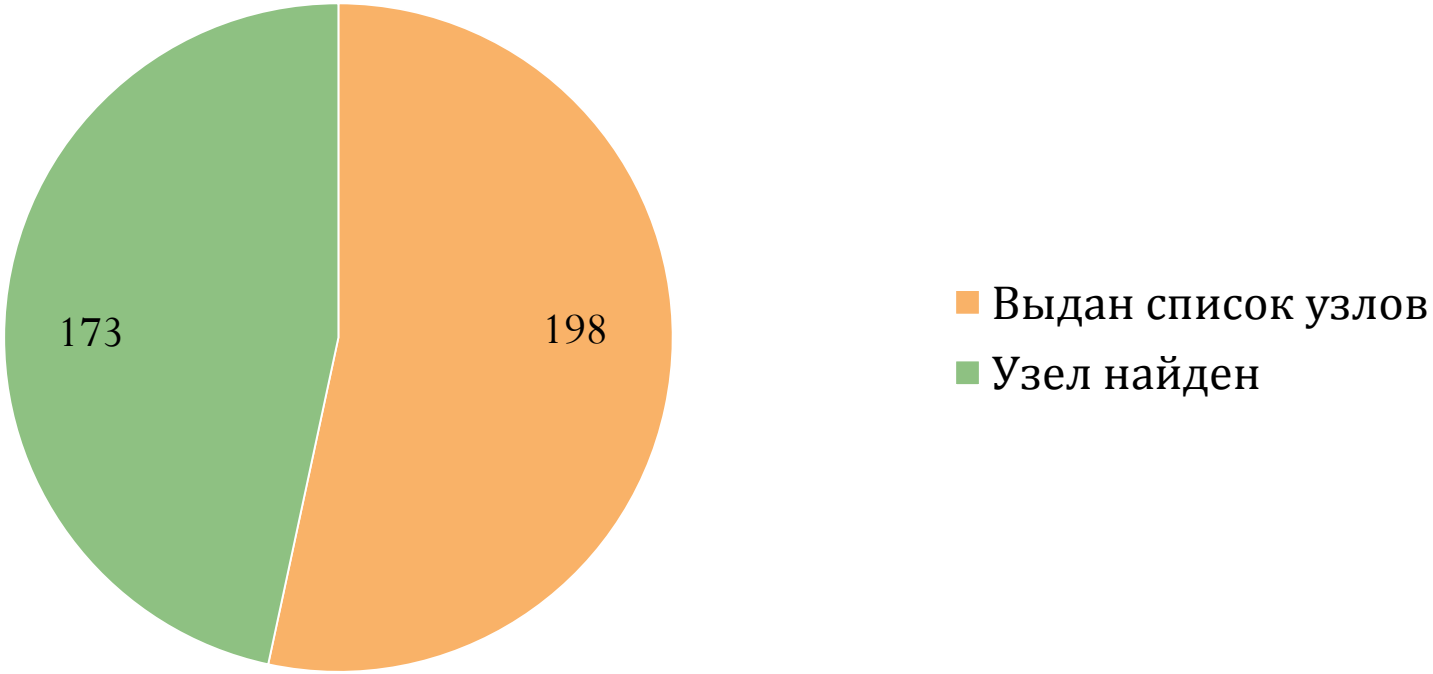


Анализ перемещенных узлов

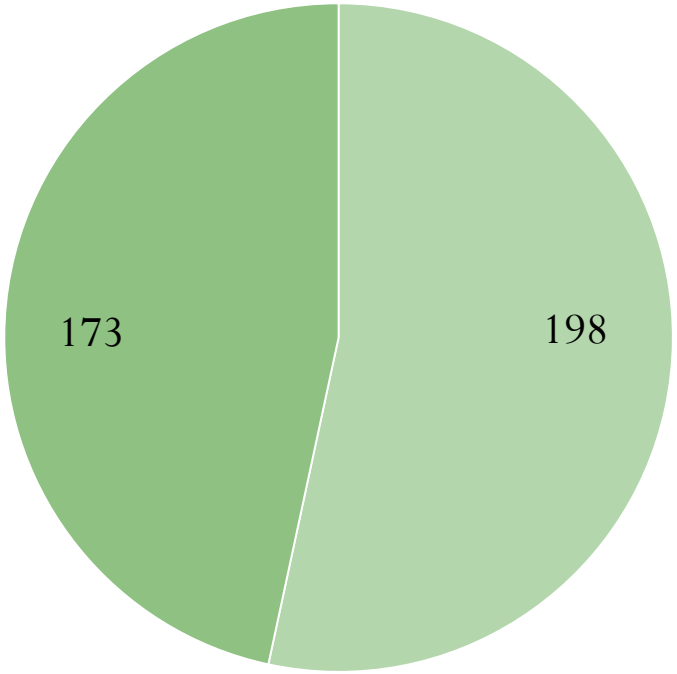


- Выдан список узлов
- Выдан список узлов (перемещенные)
- Узел найден (перемещенные)
- Узел найден

После учета перемещенных узлов

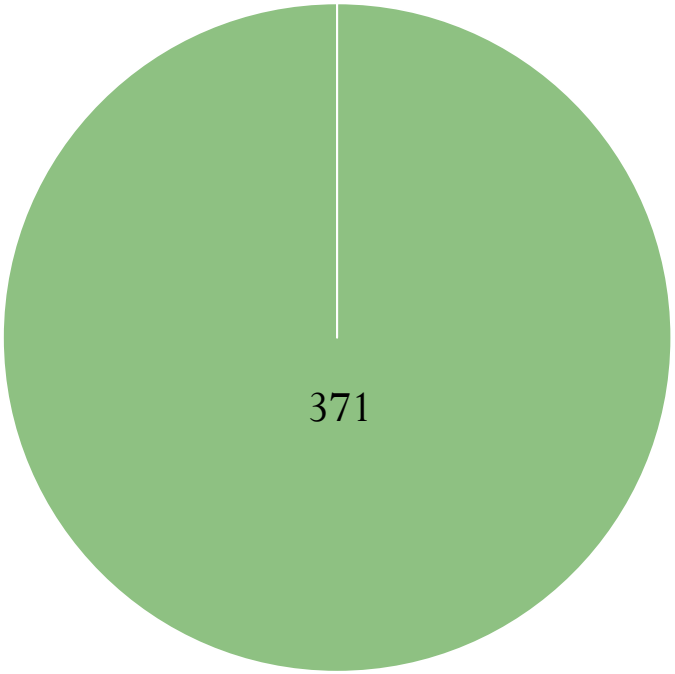


Результат поиска узлов



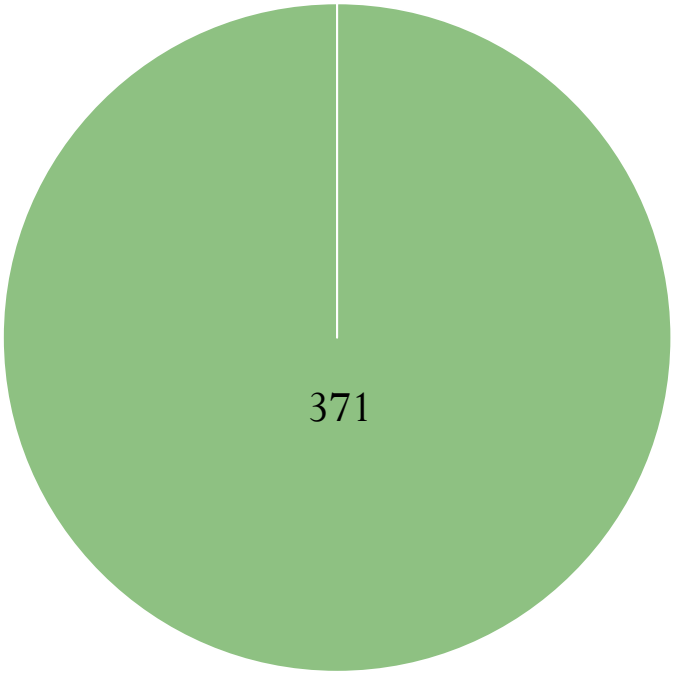
- Первый результат в списке
- Узел найден

Окончательный результат поиска узлов



Найдено безошибочно

Окончательный результат поиска узлов



Найдено безошибочно

100%

dotnet / roslyn

Watch 888 | Star 6,333 | Fork 1,515

Code | Issues 3,195 | Pull requests 125 | Projects 0 | Wiki | Pulse | Graphs

The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs.

19,572 commits | 64 branches | 49 releases | 177 contributors | Apache-2.0

dotnet / roslyn

Watch 888 | Star 6,333 | Fork 1,515

Code | Issues 3,195 | Pull requests 125 | Projects 0 | Wiki | Pulse | Graphs

The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs.

19,572 commits | 64 branches | 49 releases | 177 contributors | Apache-2.0

Initial commit — Начало анализа — 1 год разработки — Конец анализа

dotnet / roslyn

Watch 888 | Star 6,333 | Fork 1,515

Code | Issues 3,195 | Pull requests 125 | Projects 0 | Wiki | Pulse | Graphs

The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs.

19,572 commits | 64 branches | 49 releases | 177 contributors | Apache-2.0

Initial commit	Начало анализа	1 год разработки 10 340 ревизий	Конец анализа

- Изменено файлов **2 584** из 4 995
- Изменено сущностей **20 534** из 152 271 (13,5%)

dotnet / roslyn

Watch 888 | Star 6,333 | Fork 1,515

Code | Issues 3,195 | Pull requests 125 | Projects 0 | Wiki | Pulse | Graphs

The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs.

19,572 commits | 64 branches | 49 releases | 177 contributors | Apache-2.0

- Изменено сущностей: 20 534 (...много)

dotnet / roslyn

Watch 888 | Star 6,333 | Fork 1,515

Code | Issues 3,195 | Pull requests 125 | Projects 0 | Wiki | Pulse | Graphs

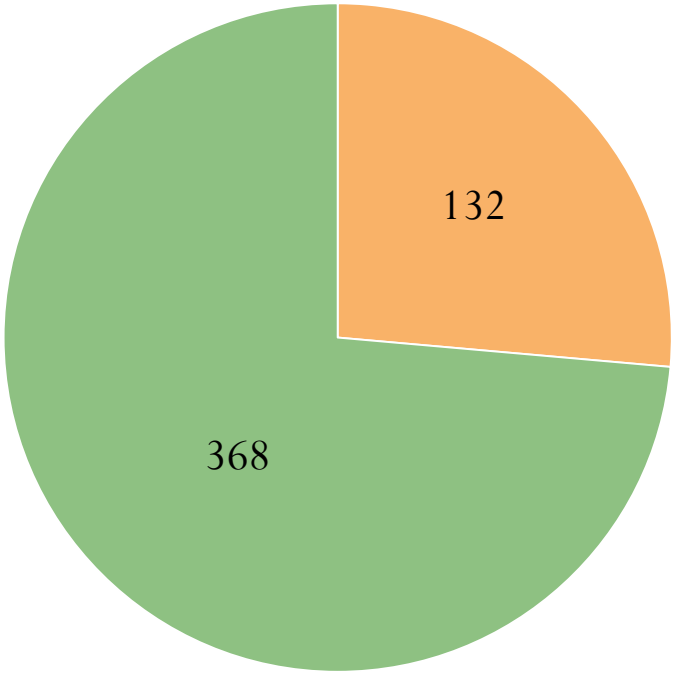
The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs.

19,572 commits | 64 branches | 49 releases | 177 contributors | Apache-2.0

- Изменено сущностей: 20 534 (...много)
- Взята выборка из 500 случайных* сущностей

* RandomShuffle(), SubList(0, 500).

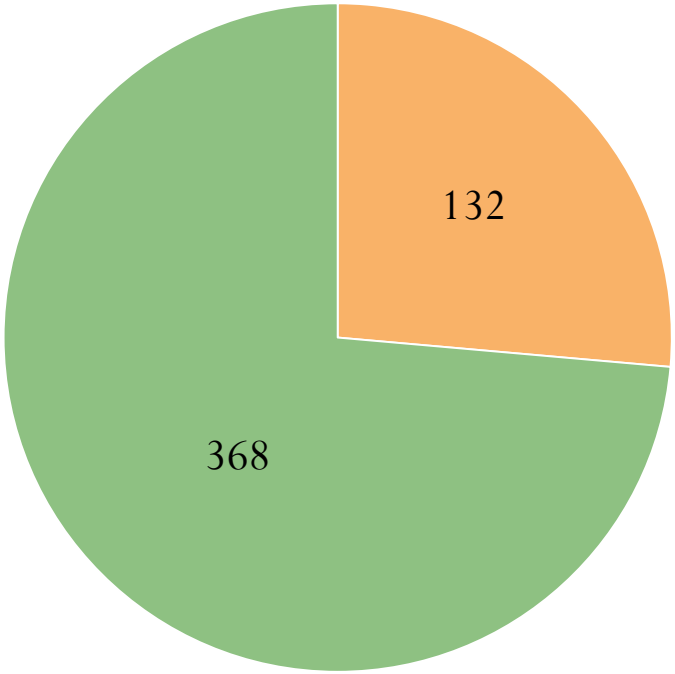
Предварительный результат поиска узлов



Выдан список узлов

Узел найден

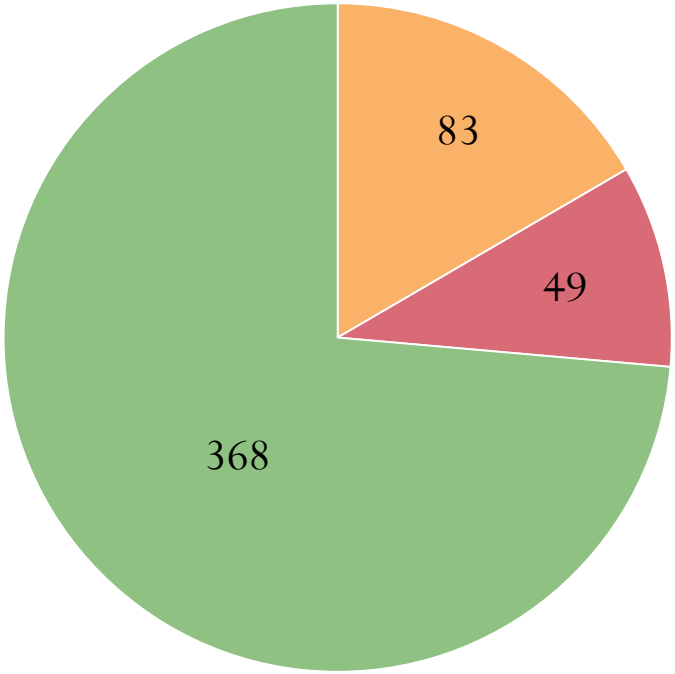
Предварительный результат поиска узлов



Выдан список узлов

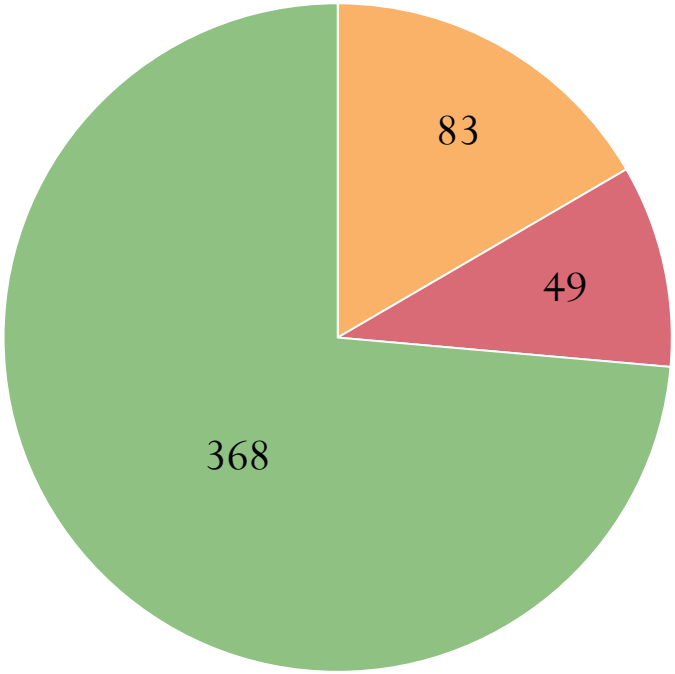
Узел найден

Удаленные узлы



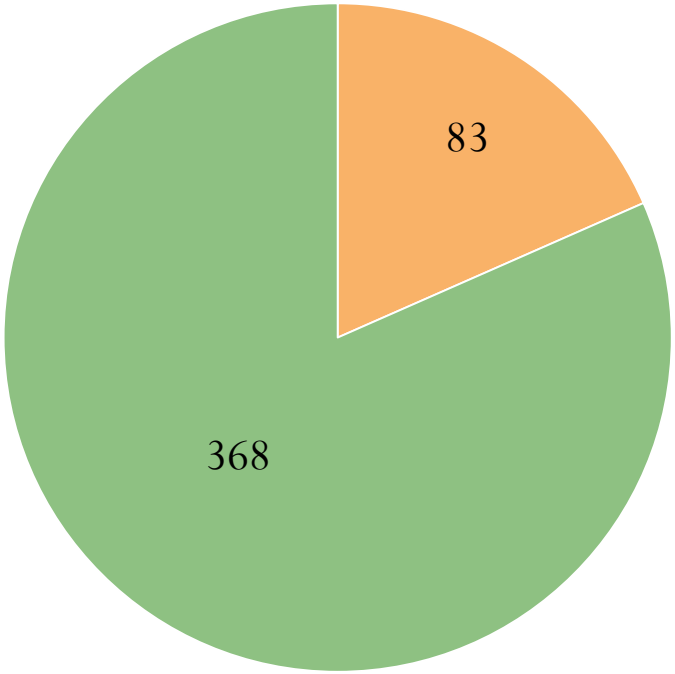
- Выдан список узлов
- Узел удален
- Узел найден

Удаленные узлы



- Выдан список узлов
- Узел удален
- Узел найден

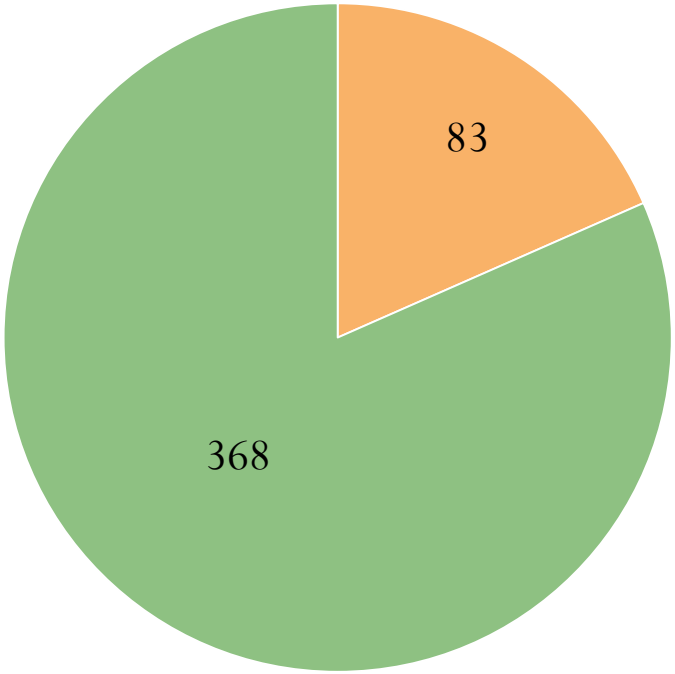
После учета удаленных узлов



Выдан список узлов

Узел найден

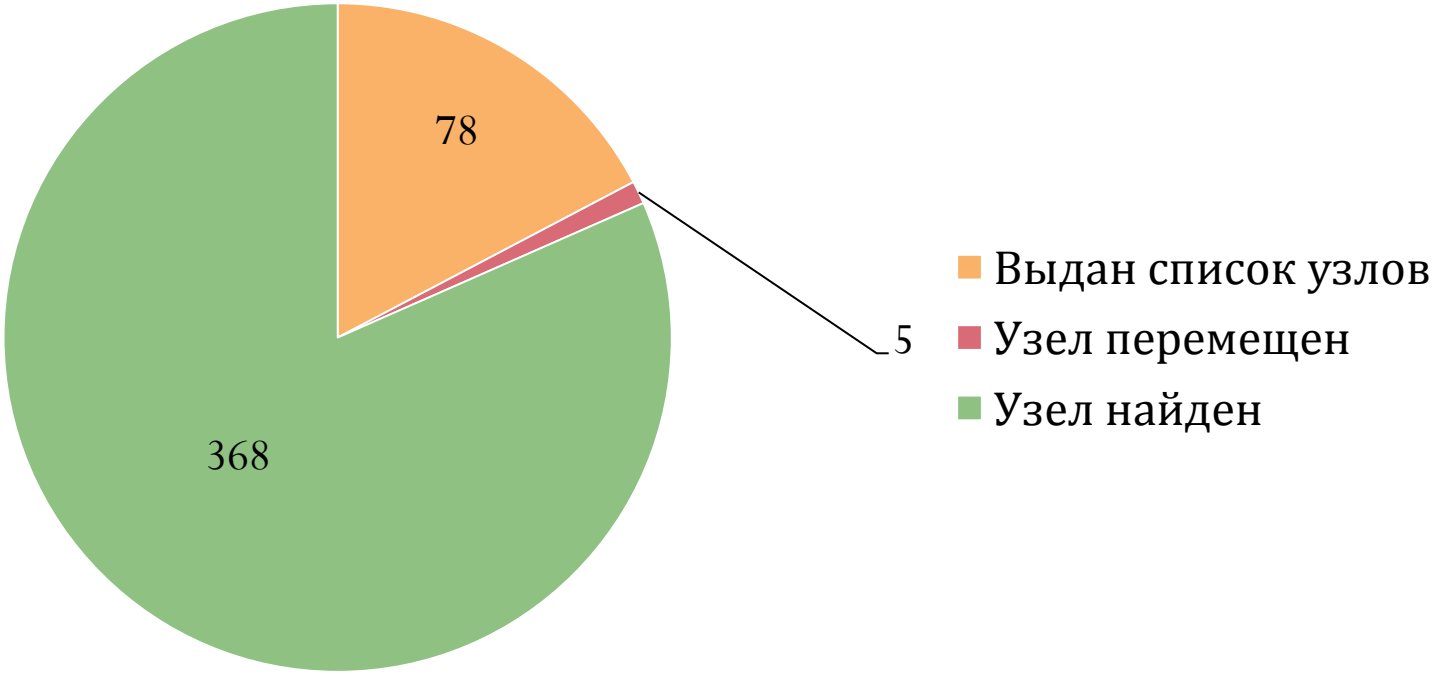
После учета удаленных узлов



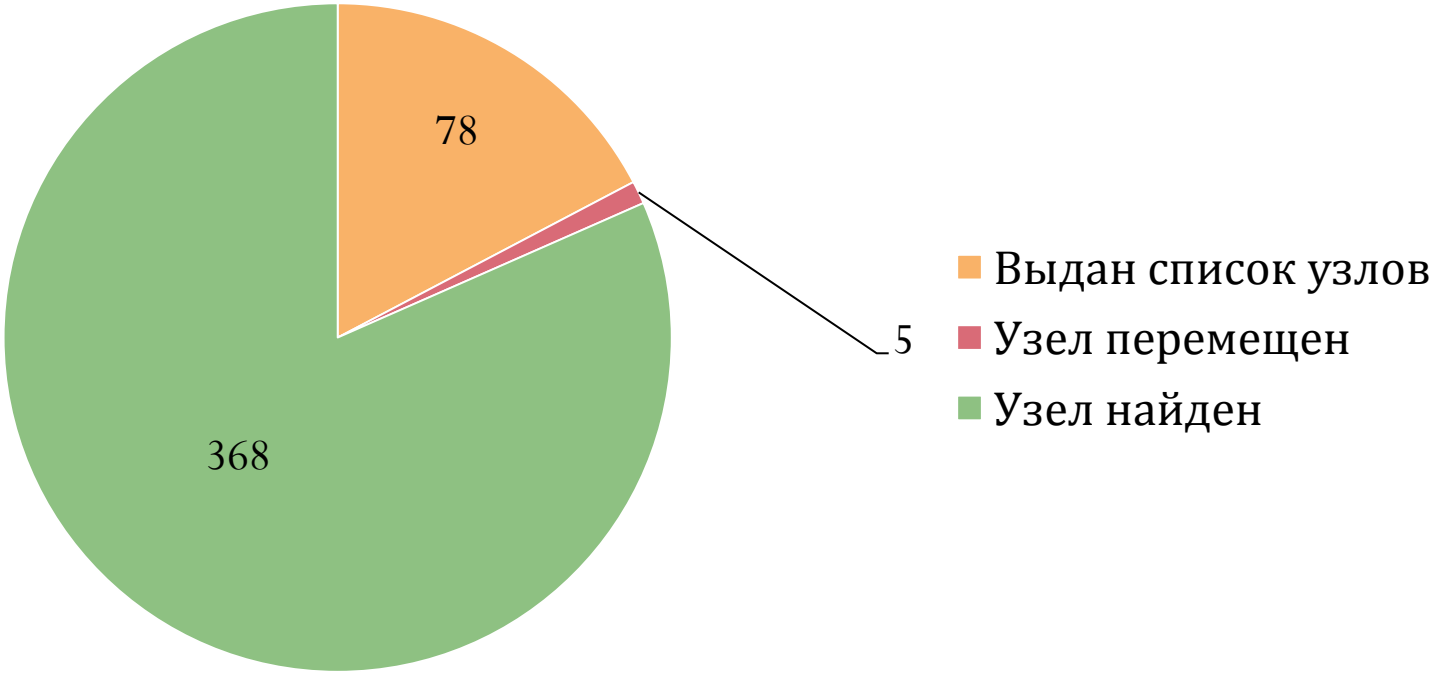
Выдан список узлов

Узел найден

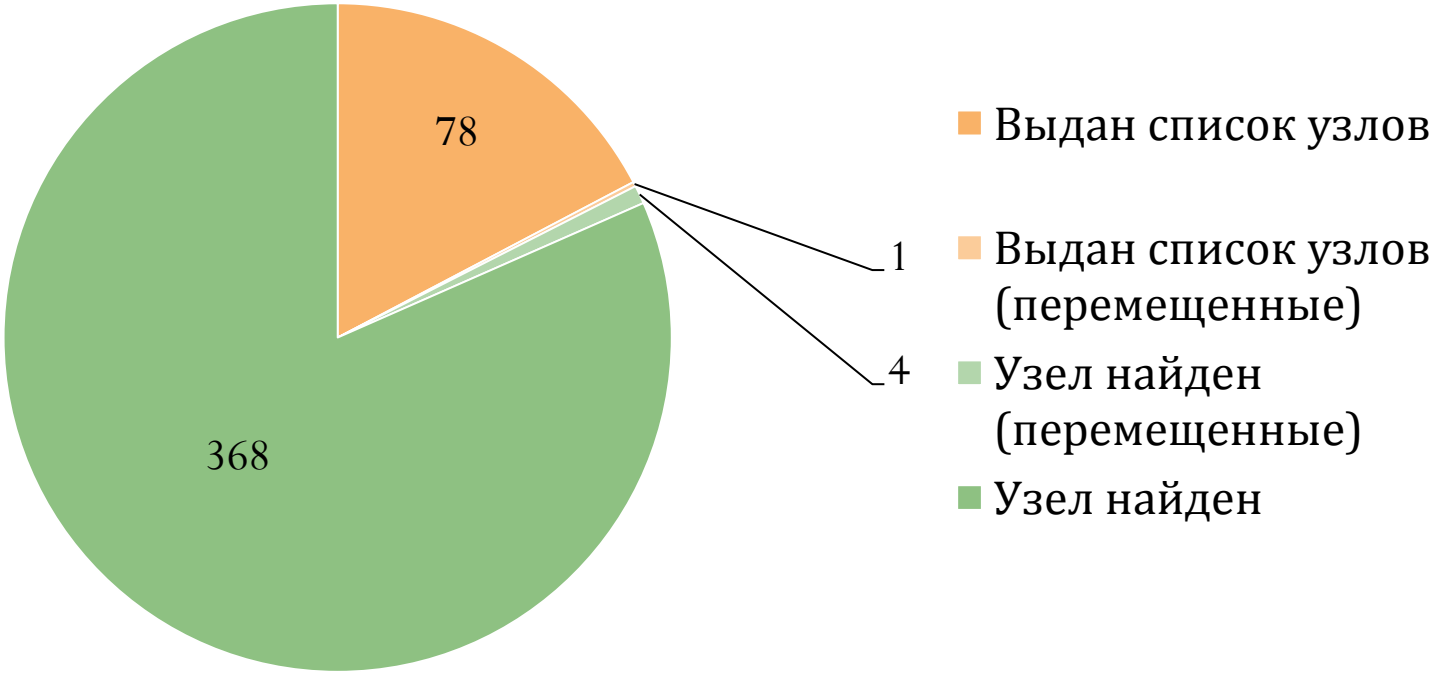
Перемещенные узлы



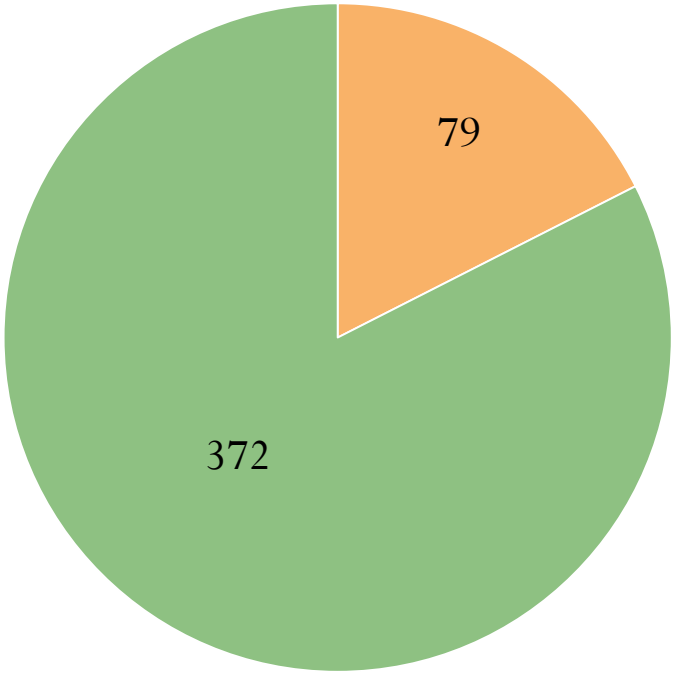
Перемещенные узлы



Анализ перемещенных узлов



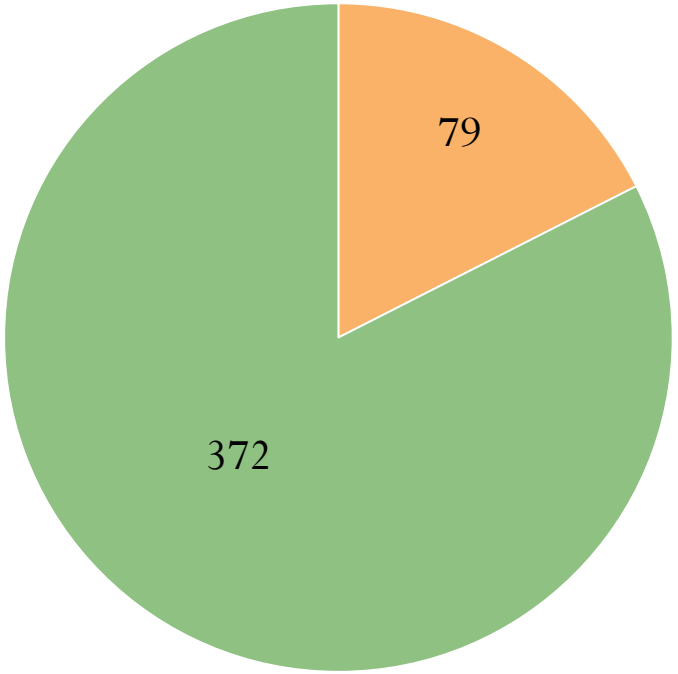
После учета перемещенных узлов



Выдан список узлов

Узел найден

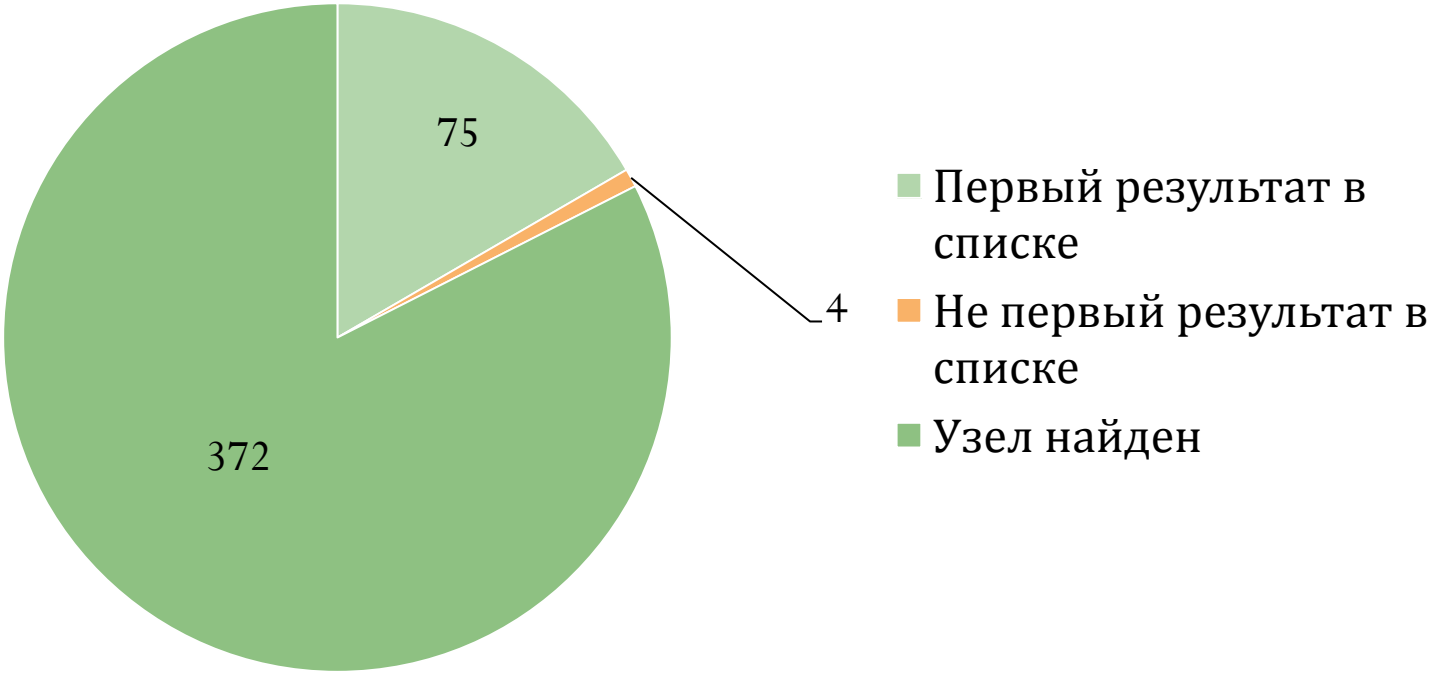
После учета перемещенных узлов



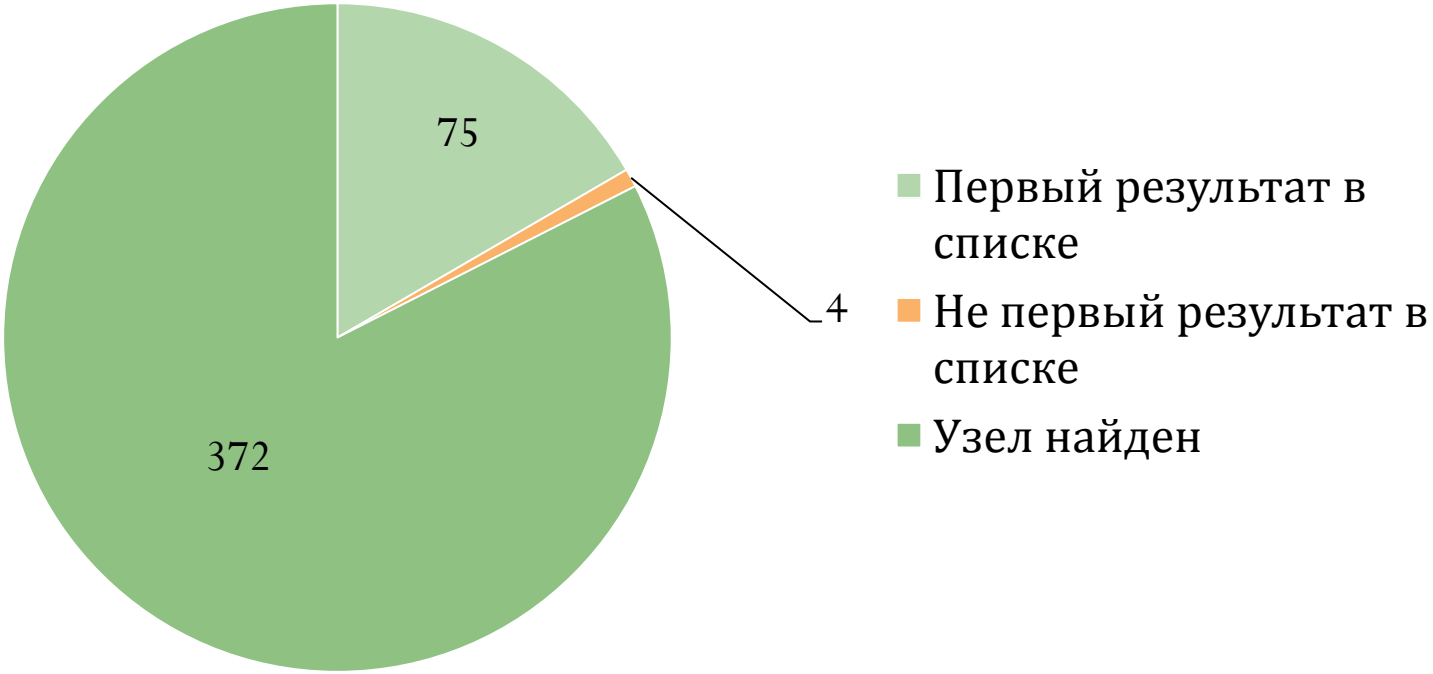
Выдан список узлов

Узел найден

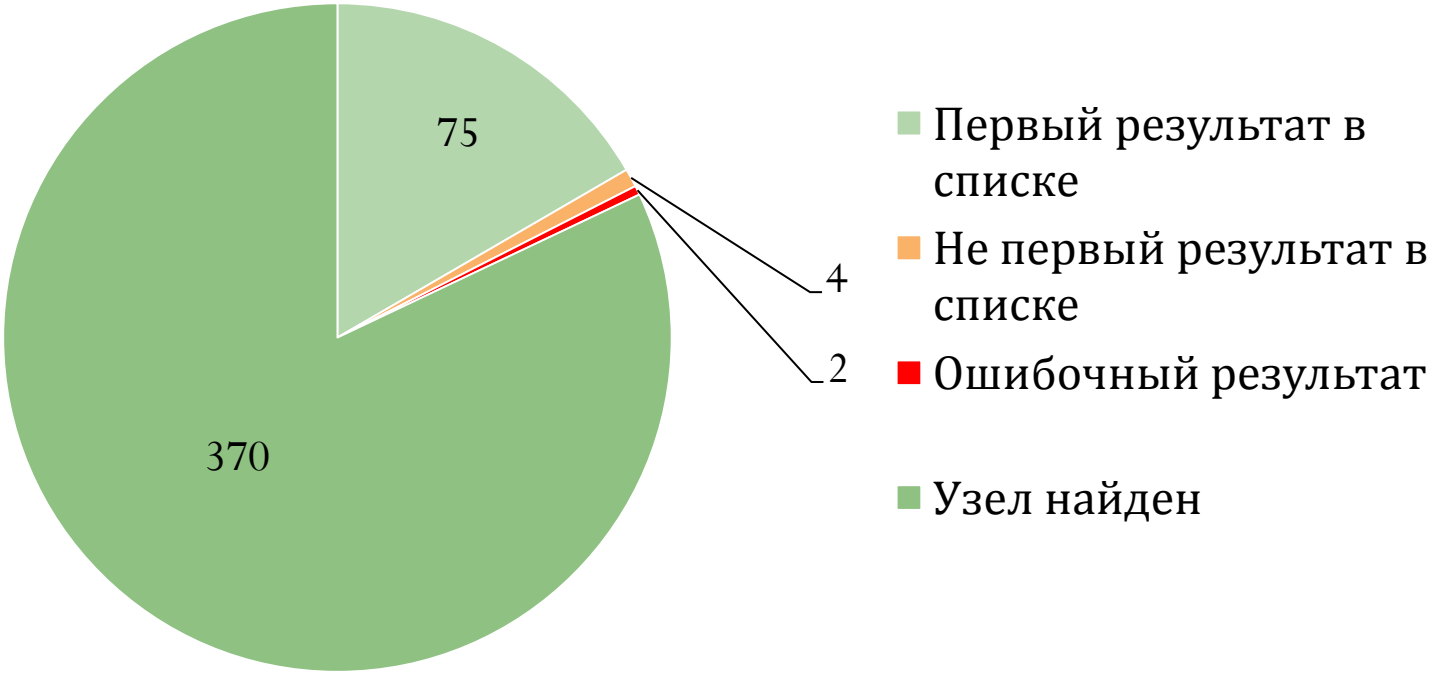
Анализ неавтоматически найденных узлов



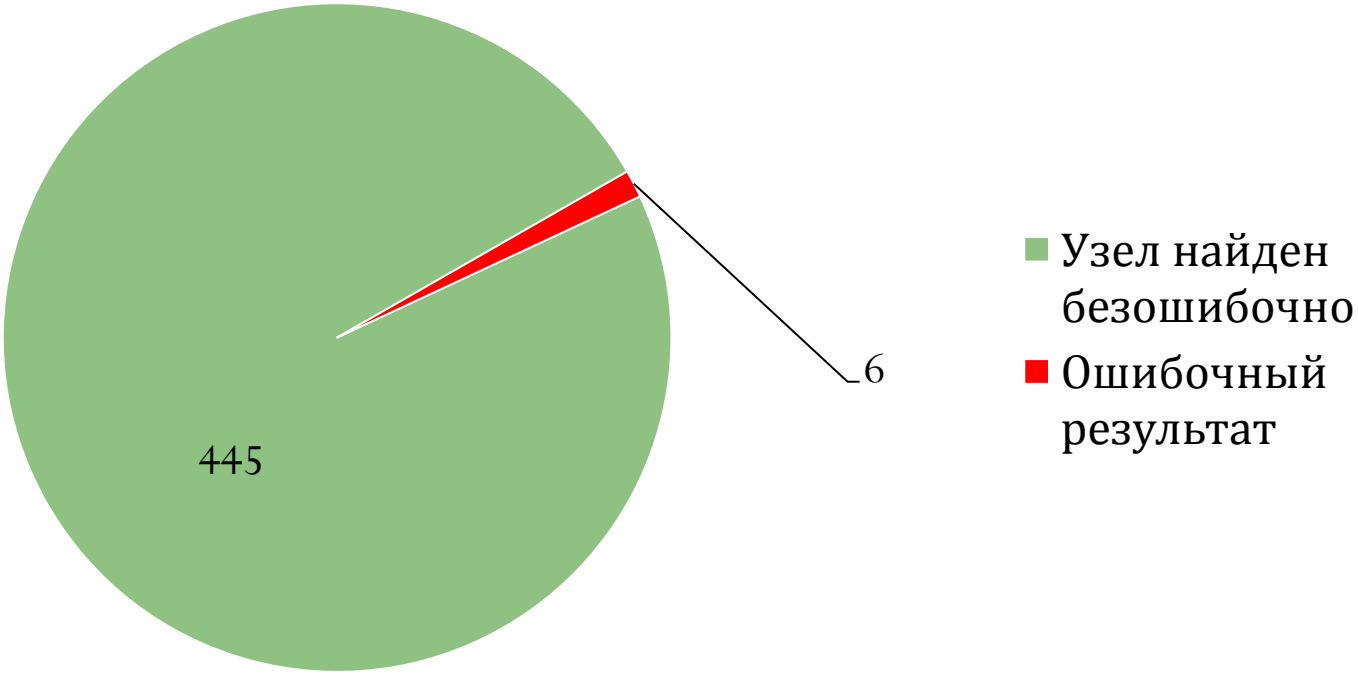
Анализ неавтоматически найденных узлов



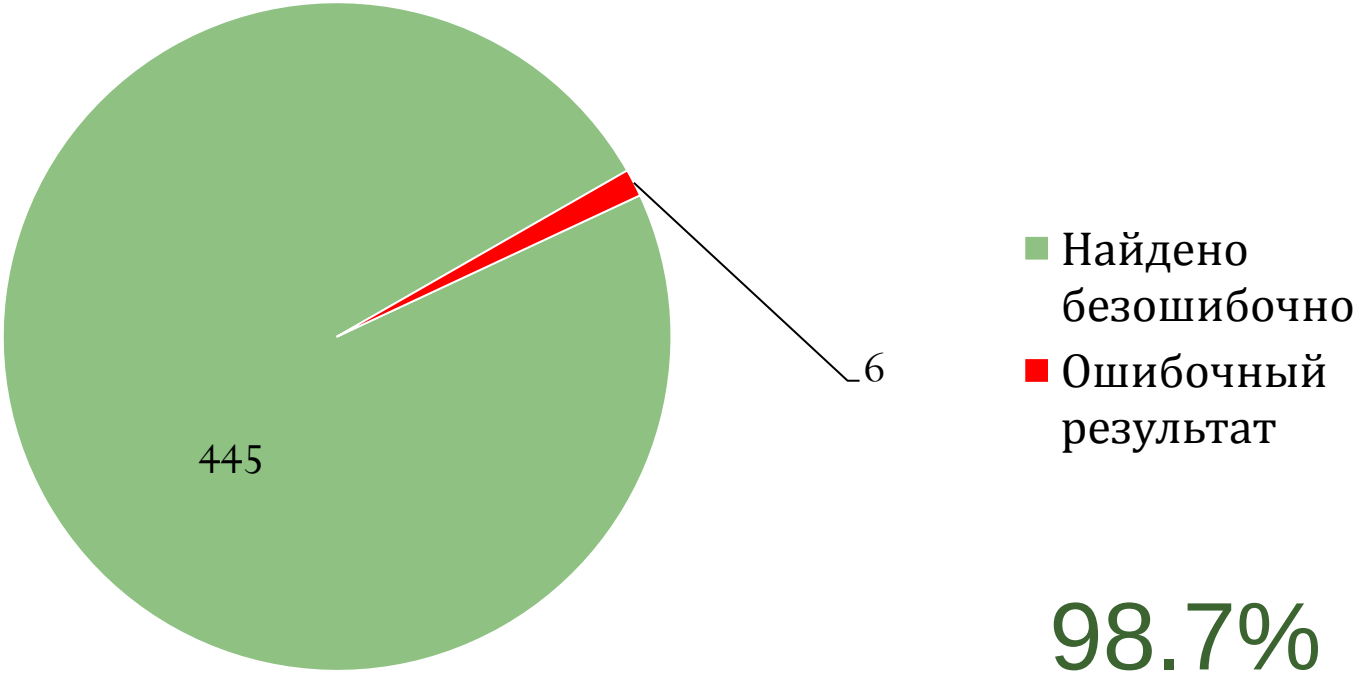
Анализ автоматически найденных узлов



Окончательный результат поиска узлов



Окончательный результат поиска узлов

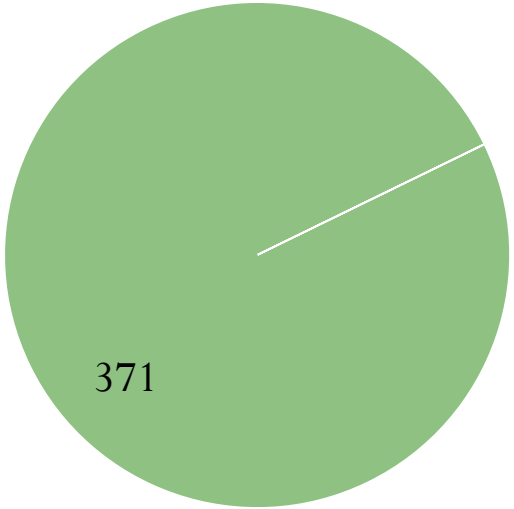


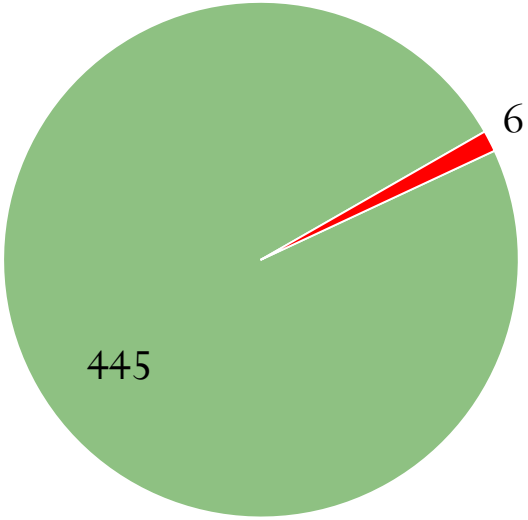
98.7%

Результат

PascalABC.NET

- 326 ревизий
- 9 месяцев



Roslyn

- 10340 ревизий
- 1 год