AI+ 软件研发数字峰会
AI+ software Development Digital summit

AI驱动软件研发
全面进入数字化时代

中国·深圳 11.24-25

AI+
software
Development
Digital
summit

基于AIGC的蚂蚁新一代测试用例自动生成技术

周海莲 蚂蚁集团

科技生态圈峰会 + 深度研习

——1000 + 技术团队的共同选择

KEYLINK ing

K+峰会

K+峰会 深圳站

K+ 全球软件研发行业创新峰会

会议时间：2024.05.24-25

K+峰会 上海站

K+ 全球软件研发行业创新峰会

会议时间：2024.09.20-21

AiDD 峰会

AiDD峰会 深圳站

AI+ 软件研发数字峰会

会议时间：2023.11.24-25

AiDD峰会 北京站

AI+ 软件研发数字峰会

会议时间：2024.07.19-20

AiDD峰会 深圳站

AI+ 软件研发数字峰会

会议时间：2024.11.15-16

演讲嘉宾

周海莲（花名：慕蓝）

蚂蚁集团平台工程与技术风险部 高级技术专家

目前主要负责测试用例自动生成方向。毕业后曾就职于百度、阿里、蚂蚁，负责过搜索系统、广告系统的质量保障和质量平台研发。

- 2020年以来加入蚂蚁，负责测试用例自动生成方向，孵化出智能单元测试用例生成产品SmartUnit，在蚂蚁集团、阿里集团的多个BU中使用并取得优秀落地效果。
- 2023年开始探索基于AIGC的新一代测试用例生成技术，并在蚂蚁内部实践落地；

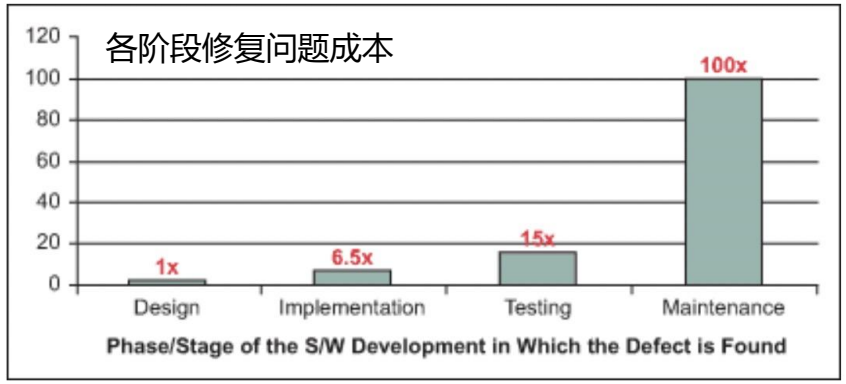
目录
CONTENTS

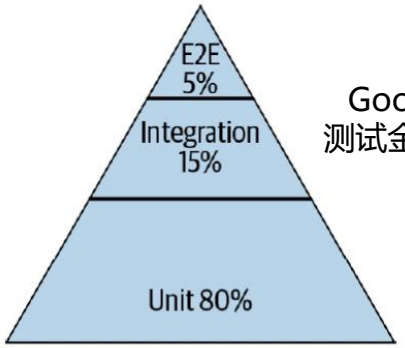
PART 01
测试用例自动生成的技术演进

测试用例自动生成

【测试智能化】 人工手写大量测试用例 —> **极致用户体验的用例编写方式：秒级智能生成高覆盖率、高有效性的测试用例**

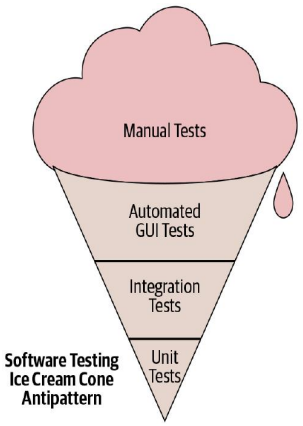
理想中的测试用例

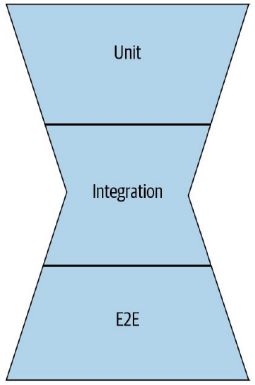




Google
测试金字塔

实际中的测试用例





实际项目中测试占比

测试用例的开发运维成本

技术演进路线

技术方向	发展历史	优势	不足
模糊测试 Fuzzing	1988年，Fuzz Generator首次被提出，用于测试Unix程序的健壮性，通过构造大量测试输入从而发现软件漏洞。 2013年，Fuzzing技术迎来分水岭：AFL，首次采用通过插桩获取代码覆盖从而引导Fuzzing的方式。这一技术随后被应用至测试用例生成领域。	Fuzzing在安全领域获得了非常好的发现漏洞效果	覆盖率不够好、生成用例量过大
基于符号执行 Symbolic Execution	1976年，Symbolic Execution的概念首次被提出，通过解析程序的路径，用符号模拟通过路径并获得输出。 2006年，研究人员提出了一种”先进行符号执行，后根据符号执行结果生成测试用例”的执行生成测试技术，这项技术后来发展出了用于检测Linux内核错误的KLEE。	对能够求解出的路径，可以精准命中分支	存在路径爆炸、约束求解复杂等问题,单一的符号执行技术难以实现测试用例生成
基于搜索 Search Based Software Testing	1990年，B. Korel提出了使用动态数据流分析的技术进行路径覆盖，这是基于搜索的测试用例生成的初始想法。 SBST从问题的解空间出发，通过启发式搜索优化算法来解决测试用例生成的问题。	可扩展性高，可以应用多种搜索算法，如遗传算法、爬山算法等等。	随机性高
基于AIGC	2023年，基于AIGC的代码生成能力惊艳亮相，给用例生成领域带来了底层核心技术的变革。	- 可读性高 - 可扩展性强 - 生成效率高 - …	生成用例的执行通过率仍在持续提升中

PART 02
现有技术的问题和痛点

用例生成的难点与挑战

技术难点

用例数据构造

复杂数据类型：大型业务系统中通常包含复杂数据类型，如多层嵌套的JSON结构、复杂Object等；

复杂语言特性：匿名内部类、lambda表达式等广泛应用于业务系统中；

用例有效性

断言语句生成：测试用例需要包含有效断言，才能具备发现问题的能力；

问题发现能力：基于代码的用例生成，断言值一般与代码保持一致。对代码问题发现有滞后性；

业务落地难点

用例可读性

机器生成的用例，由于用例语句组装策略相对固定，会导致生成的用例可读性低于人工手写用例；

多语言扩展

基于SBST、程序分析等技术进行用例生成时，多语言之间的工程化能力差异较大，扩展难度高；

用例运行稳定性

运行效率：单测用例量级较大，需保证相应的运行效率，以适配Devops；

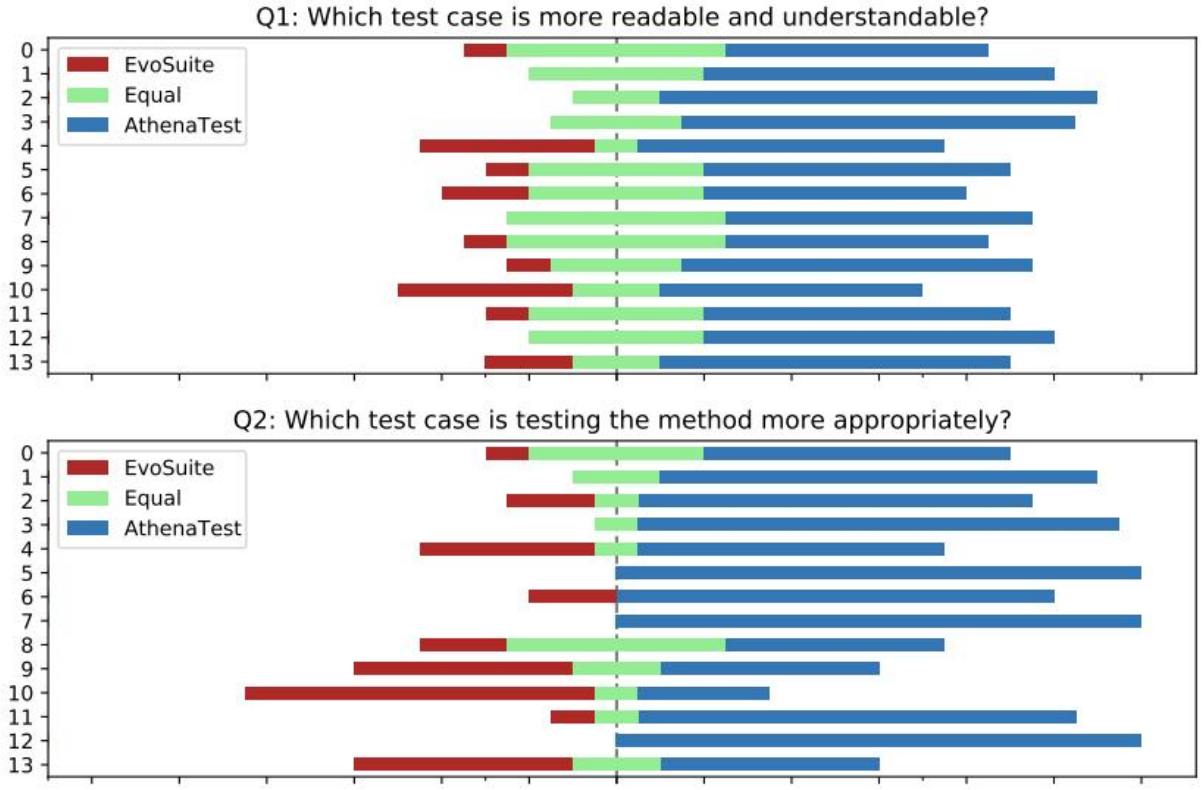
环境依赖：需进行用例执行前后的清理，以解除用例相互依赖；

AIGC浪潮下用例生成的变革

以大模型为基础进行用例生成，很多已有的难点和挑战都有了新的解决方案；

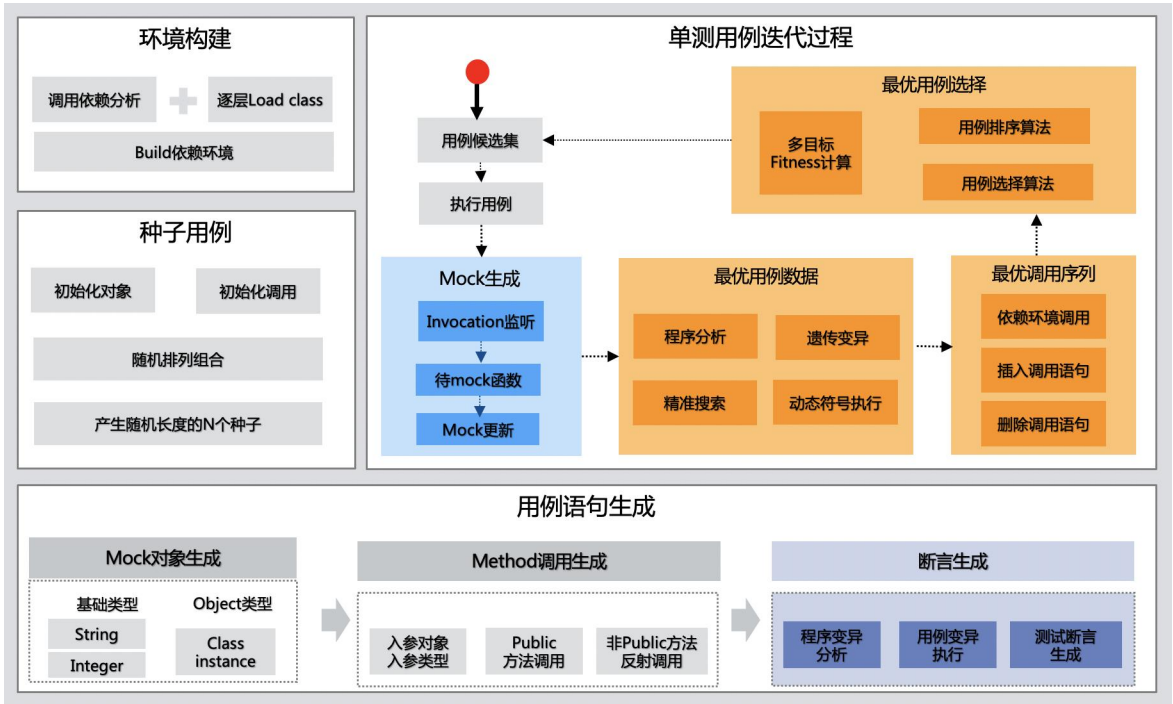
- **用例可读性**

大模型生成的测试用例可读性优于传统生成工具；



- **全语言支持**

大模型天然支持全语言，传统生成工具在跨语言支持时则会遇到很多技术难点。以SmartUnit为例，语句组装、运行时环境等都和语言紧密相关；



PART 03

基于AIGC的蚂蚁新一代测试用例自动生成

产品能力 [需求-to-测试用例]

需求-to-测试用例，根据一句话需求来生成对应的测试用例(Java & Python)。帮助研发将测试环节前置，提高问题发现效率；

- **概括性需求**：用Junit编写一个登录页面的自动化用例

以下是一个使用JUnit框架编写登录页面自动化用例的示例：

```java
import org.junit.Assert;
import org.junit.BeforeClass;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class LoginPageTest {

    private static WebDriver driver;

    @BeforeClass
    public static void setUp() {
        System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");
        driver = new ChromeDriver();
        driver.get("https://www.example.com/login");
    }

    @Test
    public void testLoginWithValidCredentials() {
        // Type valid username and password
        WebElement usernameInput = driver.findElement(By.id("username"));
        WebElement passwordInput = driver.findElement(By.id("password"));
        usernameInput.sendKeys("valid_username");
        passwordInput.sendKeys("valid_password");

        // Click login button
        WebElement loginButton = driver.findElement(By.name("loginBtn"));
        loginButton.click();

        // Check if welcome message is displayed
        WebElement welcomeMessage = driver.findElement(By.className("welcome-message"));
        Assert.assertEquals("Welcome, valid_username!", welcomeMessage.getText());
    }
```

- **聚焦性需求**：编写一个单元测试来测试一个名为hasCloseElements的方法，该方法检查输入的列表中是否有任何两个数字之间的距离小于给定的阈值

```python
import unittest
from your_module import has_close_elements, find_closest


class TestHasCloseElements(unittest.TestCase):
    def test_no_close_elements(self):
        nums = [1, 2, 3]
        threshold = 0.5
        result = has_close_elements(nums, threshold)
        expected_result = False
        self.assertEqual(result, expected_result)

    def test_one_element_list(self):
        nums = [4]
        threshold = 0.6
        result = has_close_elements(nums, threshold)
        expected_result = True
        self.assertEqual(result, expected_result)

    def test_two_elements_list(self):
        nums = [1, 7]
        threshold = 0.8
        result = has_close_elements(nums, threshold)
        expected_result = False
        self.assertEqual(result, expected_result)

    def test_three_elements_list(self):
        nums = [10, 20, 30]
        threshold = 0.9
        result = has_close_elements(nums, threshold)
        expected_result = True
        self.assertEqual(result, expected_result)

```

▶▶ 产品能力 [被测代码-to-测试用例]

被测代码-to-测试用例，根据被测代码来生成对应的测试用例，目前支持五种语言：Java、Python、JavaScript、C++、Go；

- **简单被测代码**：冒泡排序

为以下代码写单测：

```
public class BubbleSort {
 public static void bubbleSort(int[] arr) {
  int n = arr.length;
  for (int i = 0; i < n - 1; i++) {
   boolean swapped = false;
   for (int j = 0; j < n - i - 1; j++) {
    if (arr[j] > arr[j + 1]) {
     int temp = arr[j];
     arr[j] = arr[j + 1];
     arr[j + 1] = temp;
     swapped = true;
    }
   }
   if (!swapped) {
    break;
   }
  }
 }
}
```

在IDE中编辑 复制

```
import org.junit.Test;
import static org.junit.Assert.*;

public class BubbleSortTest {

    @Test
    public void testBubbleSort() {
        // Test case for sorting an array of integers in ascending order using the bubble sort algorithm
        int[] arr = new int[]{5,2,8,3};
        BubbleSort.bubbleSort(arr);
        assertArrayEquals(new int[]{2,3,5,8}, arr);

        // Test case for sorting an empty array of integers in descending order using the bubble sort algorithm
        int[] emptyArr = new int[]{};
        BubbleSort.bubbleSort(emptyArr);
        assertArrayEquals(new int[]{}, emptyArr);

        // Test case for sorting a null array of integers in ascending order using the bubble sort algorithm
        Integer[] nullArr = null;
        BubbleSort.bubbleSort(nullArr);
        assertNull(nullArr);
    }
}
```

测试数组长度为100的情况

在IDE中编辑 复制

```
@Test
public void testBubbleSortWithLargeInputSize(){
    int[] inputArr = new int[100];
    Random rand = new Random();
    for(int i=0;i<inputArr.length;i++){
        inputArr[i]=rand.nextInt(101);
    }
    BubbleSort.bubbleSort(inputArr);
    Arrays.sort(inputArr);
    assertEquals("The output of the function should be a sorted array", Arrays.toString(inputArr), Arrays.toString(BubbleSort.
}

这个测试用例会验证当输入数组长度超过100时，排序函数是否按预期工作。我们使用Java中的Random类来随机生成一个包含100个元素的整数数组，并将其传递给排序函
```

产品能力 [被测代码-to-测试用例]

- **复杂被测代码**：包含外部http请求

为以下代码写单测：

```
public JSONArray queryTree(String [illegible]) {
    RequestBuilder builder = new Reque[illegible]Builder()
        .addQueryParam("[illegible]es")
        .addQueryParam("[illegible]
        .addQueryParam("[illegible]
        .addQueryParam("[illegible]
        .addQueryParam("[illegible]
        .setUrl(daYuApi);
    int times = 0;
    while (times < 3) {
      try {
        ListenableFuture<Response> future = httpclient.executeRequest(builder);
        Response response = future.get();

        JSONObject responseBody = JSONObject.parseObject(response.getResponseBody());
        int statusCode = response.getStatusCode();
        String errorMsg;
        if (statusCode == 200 && responseBody.getBoolean("success")) {
          return JSONObject.parseObject(response.getResponseBody()).getJSONArray("result");
        } else if (statusCode >= 300 && statusCode < 400) {
          errorMsg = response.getHeader("Location");
        } else {
          errorMsg = response.getResponseBody();
        }
        log.error("query trace failed[illegible]d,
            statusCode, errorMsg);
      } catch (Exception e) {
        log.error("query trace failed! [illegible]
      }
      try {
        times++;
        Thread.sleep(1000L * times);
      } catch (Exception e) {
        log.error(e.getMessage(), e);
      }
    }
    return null;
  }
```

```
@Test
void testQueryTree() throws Exception {
    // 准备数据
    String [illegible]Id = "123456";
    JSONArray expectedResult = new JSONArray().fluentAdd("[{\"arn\":\"test-arn\",\"data\":[],\"dataType\":\"\","
            + "\"idc\":\"test-idc\",\"ldc\":\"test-ldc\",\"name\":\"test-name\",\"nodeType\":\"service\","
            + "\"period\":0,\"site\":\"test-site\",\"status\":\"online\",\"type\":\"test-type\"}]");

    // mock依赖项
    EasyMock.expect(httpclient.executeRequest((RequestBuilder) EasyMock.anyObject())).andReturn(new ListenableFuture<Response>() {
        @Override
        public boolean cancel(boolean mayInterruptIfRunning) {
            return false;
        }

        @Override
        public boolean isCancelled() {
            return false;
        }

        @Override
        public boolean isDone() {
            return true;
        }

        @Override
        public Response get() throws InterruptedException, ExecutionException {
            Response response = new Response();
            response.setStatusCode(200);
            JSONObject responseBody = new JSONObject();
            responseBody.put("success", true);
            responseBody.put("result", expectedResult);
            response.setResponseBody(responseBody.toJSONString());
            return response;
        }

        @Override
        public Response get(long timeout, TimeUnit unit) throws InterruptedException, ExecutionException, TimeoutException {
            return get();
        }
    });

    EasyMock.replay(httpclient);

    // 执行测试
    JSONArray actualResult = service.queryTree([illegible]Id);
```

产品能力 [需求-测分-测试用例]

需求-测分-测试用例：根据需求描述生成对应的测试场景，再结合代码生成最终的测试用例，使得测试用例的校验能力与需求匹配，用于检查代码实现的逻辑错误；

我有一段代码，输入是userStaffNo和belongingWorkbenchCode，
在zcache中查询是否存在对应的用户权限结果，如果有返回查询结果，如果没有返回null。
请列出这个功能的测试场景

1. 输入正确的userStaffNo和belongingWorkbenchCode，预期返回查询到的用户权限结果；
2. 输入错误的userStaffNo或belongingWorkbenchCode，预期返回null；
3. 输入空值作为参数，预期返回null；
4. 输入不存在的userStaffNo和belongingWorkbenchCode，预期返回null。

以下是代码实现，结合上述测试场景，请为该代码写单测

```
public class UserRolePermissionFacadeImpl implements UserRolePermissionFac
private UserRolePermissionResult queryInCache(String userStaffNo, String b
        String key =
        String value = zcacheService.getValue(key);

        UserRolePermissionResult userRolePermissionResult = null;
        if (StringUtils.isNotBlank(value)) {
            userRolePermissionResult = JSON.parseObject(value, UserRolePer
        }
        return userRolePermissionResult;
    }
    }
```

在IDE中编辑 复制

```
@Tested
private UserRolePermissionFacadeImpl userRolePermissionFacade;

// 测试输入正确的userStaffNo和belongingWorkbenchCode，预期返回查询到的用户权限结果
@Test
public void testQueryInCacheWithCorrectParams() {
    UserRolePermissionResult expectedResult = new UserRolePermissionResult();
    expectedResult.setUser("test_user");
    expectedResult.setBelongingWorkbenchCode("test_workbench");
    expectedResult.setRoles(Lists.newArrayList());

    when(zcacheService.getValue(any())).thenReturn(JSON.toJSONString(expectedResult));

    UserRolePermissionResult actualResult = userRolePermissionFacade.queryInCache("123456", "test_workbench");

    assertEquals(expectedResult, actualResult);
}

// 测试输入错误的userStaffNo或belongingWorkbenchCode，预期返回null
@Test
public void testQueryInCacheWithIncorrectParams() {
    assertNull(userRolePermissionFacade.queryInCache("", ""));
    assertNull(userRolePermissionFacade.queryInCache("invalid_staff_no", ""));
    assertNull(userRolePermissionFacade.queryInCache("valid_staff_no", "invalid_workbench"));
}

// 测试输入空值作为参数，预期返回null
@Test
public void testQueryInCacheWithEmptyParam() {
    assertNull(userRolePermissionFacade.queryInCache("", ""));
    assertNull(userRolePermissionFacade.queryInCache("", "valid_workbench"));
}

// 测试输入不存在的userStaffNo和belongingWorkbenchCode，预期返回null
@Test
public void testQueryInCacheWithNonexistentParams() {
    assertNull(userRolePermissionFacade.queryInCache("nonexistent_staff_no", "nonexistent_workbench"));
    assertNull(userRolePermissionFacade.queryInCache("valid_staff_no", "nonexistent_workbench"));
}
```

产品能力 [测试用例补全]

测试用例补全：对存量未包含Assert的测试用例进行补全，增强用例有效性；

下面是被测代码

```
public static int hexDigitToInt(final char hexDigit) {
    final int digit = Character.digit(hexDigit, 16);
    if (digit < 0) {
        throw new IllegalArgumentException("Cannot interpret '" + hexDigit + "' as a hexadecimal
digit");
    }
    return digit;
}
```

下面代码是针对上面被测代码生成的用例,请补全用例,生成assert校验

```
@Test
public void testHexDigitToInt_withValidInput() {
    char hexDigit = 'A';
    int result = Conversion.hexDigitToInt(hexDigit);
}

@Test
public void testHexDigitToInt_withInvalidInput() {
    char hexDigit = 'Z';
    Conversion.hexDigitToInt(hexDigit);
}
```

测试用例GPT

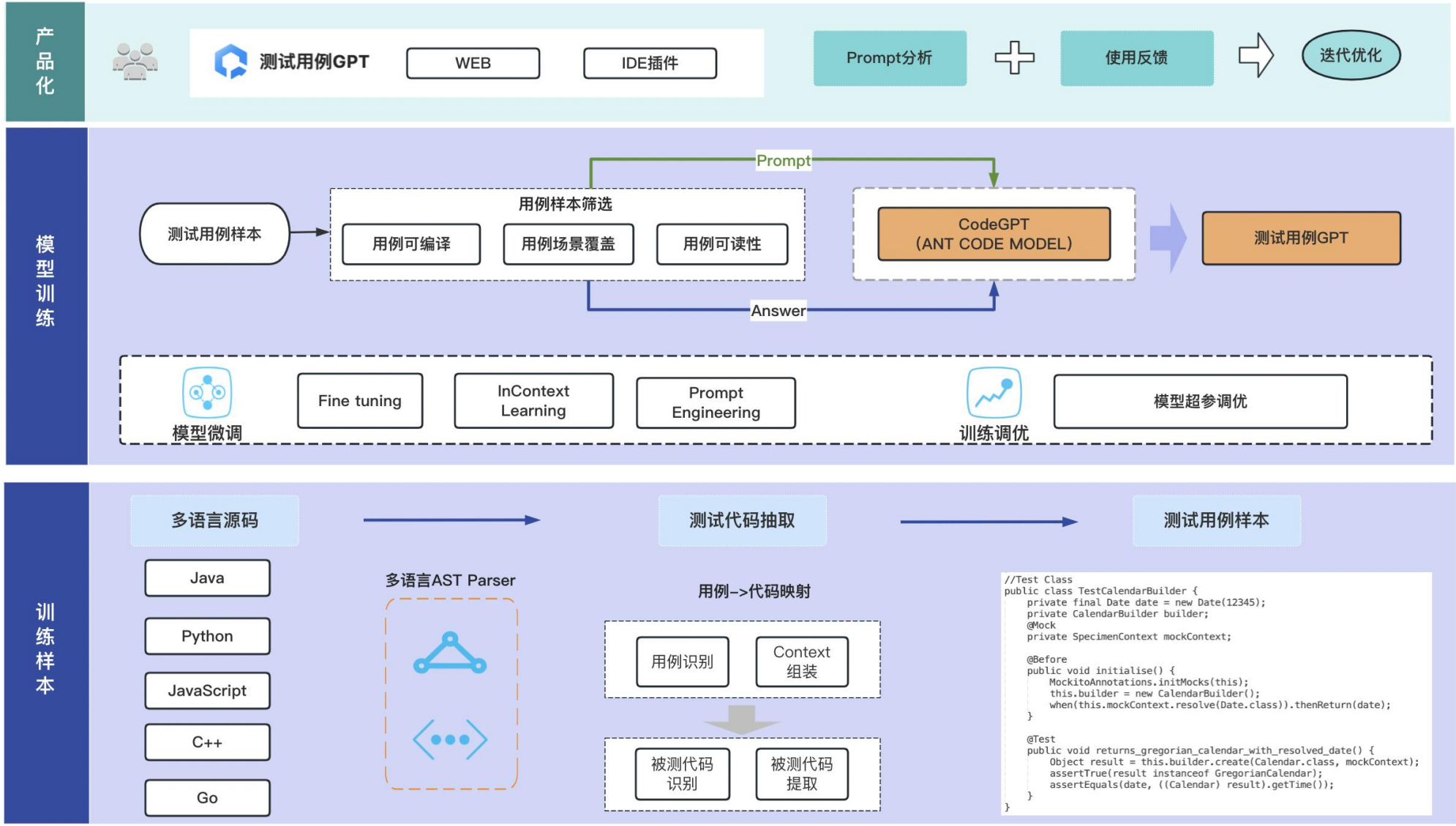
以下是生成结果：

```
1  public class ConversionTest {
2      @Test
3      public void testHexDigitToInt_withValidInput() {
4          char hexDigit = 'A';
5          int result = Conversion.hexDigitToInt(hexDigit);
6          Assert.assertEquals(10, result);
7      }
8
9      @Test(expected = IllegalArgumentException.class)
10     public void testHexDigitToInt_withInvalidInput() {
11         char hexDigit = 'Z';
12         Conversion.hexDigitToInt(hexDigit);
13     }
```

以开源代码仓库commons-lang中的代码为基础，设计评测集。以pass@1和Assert强校验占比作为评测指标：

指标	说明	指标值
pass@1	自动补充Assert后，用例正常执行并且Assert校验通过	67.7%
Assert强校验占比	自动补充的Assert为强校验的占比	97.8%

技术大图



测试用例训练样本

多语言测试用例样本抽取框架，累计抽取原始样本量约800W；

测试生成模型训练

- ✓ 需求-to-测试用例
- ✓ 被测代码-to-测试用例
- ✓ 需求-测分-测试用例
- ✓ 测试用例补全

产品化

- ✓ WEB Chat
- ✓ IDE插件
- ✓ CI组件

高质量样本构建

模型训练使用的数据质量对效果至关重要，如何定义高质量测试用例训练数据?

定义

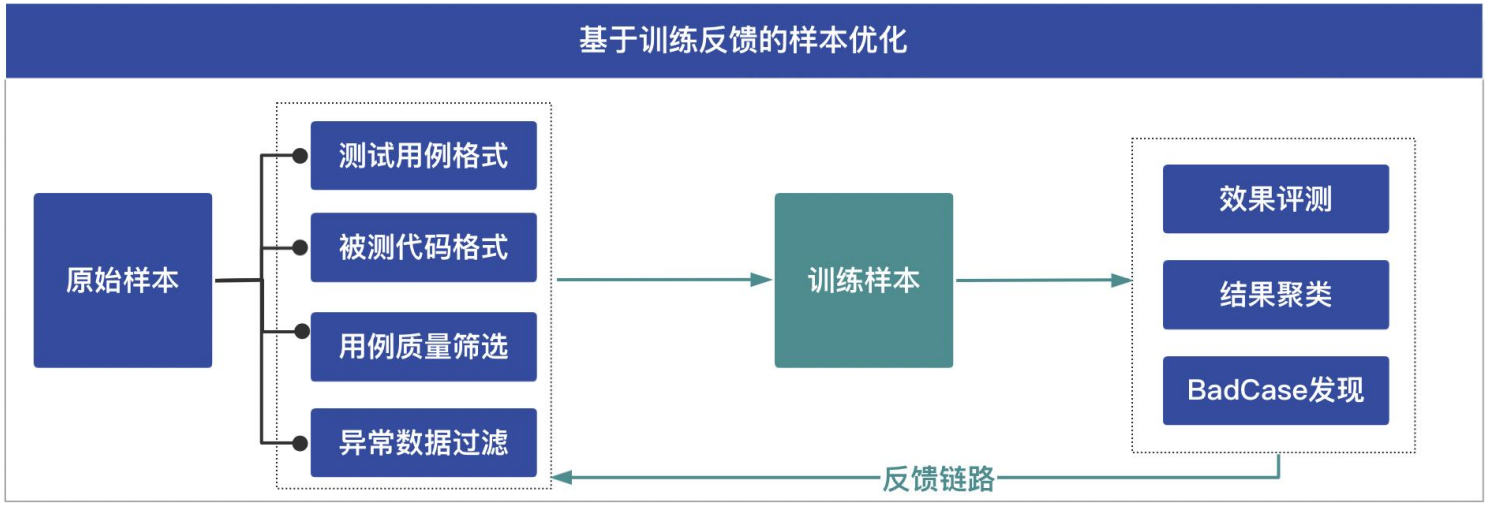
高质量训练数据定义：

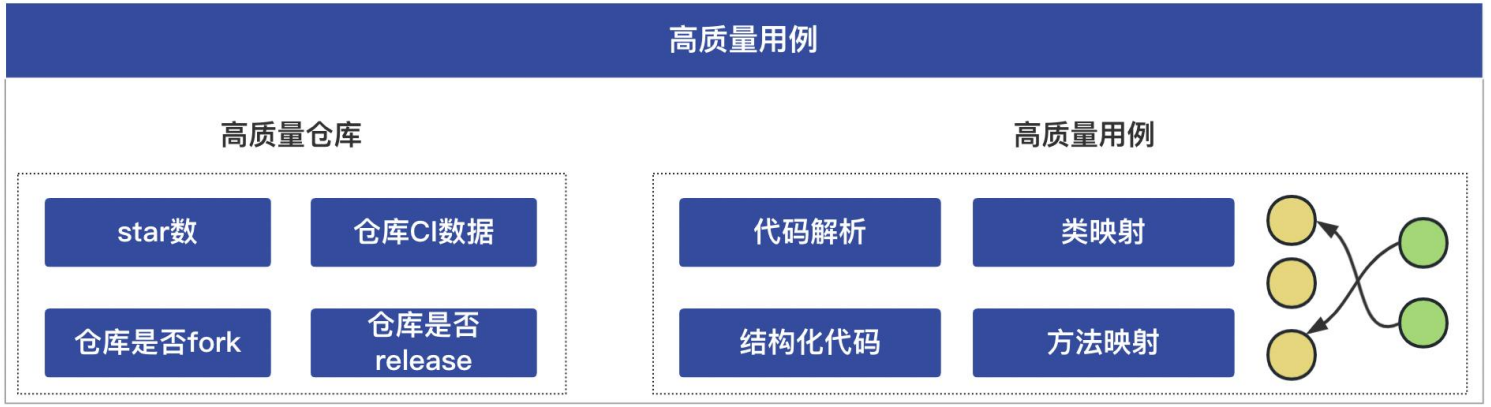
- 测试用例与被测代码精确匹配
- 代码语法正确
- 代码编译通过
- 测试用例运行通过

探索

高质量用例：

- 代码仓库Meta信息，如star数、是否fork、单测用例量级；
- 代码仓库CI信息，最近一次CI与Coverage上报状态；
- 代码分支与用例场景的映射；





模型效果评测

代码生成类任务评测集

评测集	评测集介绍
HumanEval	HumanEval由OpenAI提出。Codex构建了一个包括164个人工手写的Python编程问题的数据集HumanEval，其中每个编程问题包括函数头、docstrings、函数体和几个 unit tests。HumanEval中的编程问题可以用来评估语言理解能力、推理能力、算法能力和简单的数学能力。
HumanEval-X	由于HumanEval只支持Python语言，因此智谱提出了HumanEval-X来更好地评测代码生成模型的多语言生成能力。HumanEval-X包含820个高质量手写样本，覆盖Python、C++、Java、JavaScript、Go，可用于多种任务。
MBPP	Google提出MBPP评测集（Mostly Basic Programming Problems），共包含974个Python编程任务
CodeXGLUE	微软发布的代码智能领域的基准数据集（CodeXGLUE），包含code-code、code-text、text-code、text-text 四个类别

代码生成类任务评测指标

指标名	指标介绍
BLEU	计算生成代码和标准答案代码之间 N-gram 的匹配程度。
CodeBLEU	在BLEU之上，CodeBLEU 还要同时考虑代码的关键字信息、AST 结构化子树信息以及代码变量之间的数据流信息。CodeBLEU 的最终结果由各个子部分加权求和获得；
pass@k	每个问题生成了k个解法。如果k个解法中任一个通过了全部单元测试，则认为这个问题被解决了。 pass@k则是全部问题被解决的比例。 一般关注pass@1；

模型效果评测

类比代码生成，使用HumanEval-X评测集进行评估，核心指标采用pass@1；

task_id (string)	prompt (string)	declaration (string)	canonical_solution (string)	test (string)	example
"Java/0"	"import java.util.*; import java.lang.*; class Solution { /** Check if in given list of numbers, are any two numbers closer to each other than given threshold. >>> hasCloseElements(Arrays.asList(1.0, 2.0, 3.0), 0.5) false >>> hasCloseElements(Arrays.asList(1.0, 2.8, 3.0, 4.0, 5.0, 2.0), 0.3) true */ public boolean hasCloseElements(List<Double> numbers, double threshold) { "	"import java.util.*; import java.lang.*; class Solution { public boolean hasCloseElements(List<Double> numbers, double threshold) { "	" for (int i = 0; i < numbers.size(); i++) { for (int j = i + 1; j < numbers.size(); j++) { double distance = Math.abs(numbers.get(i) - numbers.get(j)); if (distance < threshold) return true; } } return false; } }"	"public class Main { public static void main(String[] args) { Solution s = new Solution(); List<Boolean> correct = Arrays.asList(s.hasCloseElements(new ArrayList<> (Arrays.asList(11.0, 2.0, 3.9, 4.0, 5.0, 2.2)), 0.3), !s.hasCloseElements(new ArrayList<> (Arrays.asList(1.0, 2.0, 3.9, 4.0, 5.0, 2.2)), 0.05), s.hasCloseElements(new ArrayList<> (Arrays.asList(1.0, 2.0, 5.9, 4.0, 5.0)), 0.95), !s.hasCloseElements(new ArrayList<> (Arrays.asList(1.0, 2.0, 5.9, 4.0, 5.0)), 0.8), s.hasCloseElements(new ArrayList<> (Arrays.asList(1.0, 2.0, 3.0, 4.0, 5.0, 2.0)), 0.1), s.hasCloseElements(new ArrayList<> (Arrays.asList(1.1, 2.2, 3.1, 4.1, 5.1)), 1.0), !s.hasCloseElements(new ArrayList<> (Arrays.asList(1.1, 2.2, 3.1, 4.1, 5.1)), 0.5)); if (correct.contains(false)) { throw new	"public main(St List<Bo !s.hasC (Arrays s.hasCl (Arrays); if (Asserti

- **测试用例生成评测数据集构建**

prompt使用HumanEval-X中的declaration + canonical_solution，针对prompt调用模型生成测试用例类（包含多个tests、多个assert），测试用例类中的全部tests都执行通过则认为pass；

- **模型效果**

使用HumanEval-X评测集对模型的测试用例生成能力进行评估（下表中为7b的评测结果，13b模型pass@1模型效果更优）

Java pass@1	Python pass@1	Js pass@1	Go pass@1	C++ pass@1
48.6%	35.67%	39.02%	22.1%	56.2%

TestGPT-7B模型现已开源，欢迎大家围观试用：

https://github.com/codefuse-ai/Test-Agent

PART 04
总结与展望

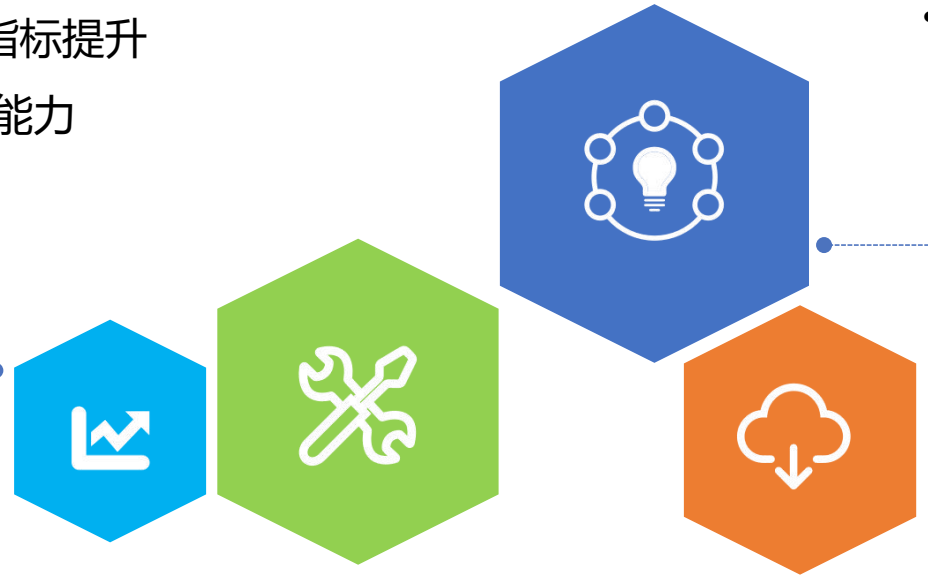
总结与展望

模型能力优化

- pass@k等核心指标提升
- 知识融会贯通的能力

丰富产品场景

- 以研发活动中质量各个环节为切入点，持续丰富场景



模型工程化能力

- 用例prompt自动组装
- 用例全生命周期管理

开源共建

- 开放共建、快速迭代

AiDD AI+ 软件研发数字峰会
AI+ software Development Digital summit

THANKS