MICROOH 麦可网

Android-从程序员到架构师之路

出品人：Sundy
讲师：高焕堂（台湾）

http://www.microoh.com

C07_f

问题集：

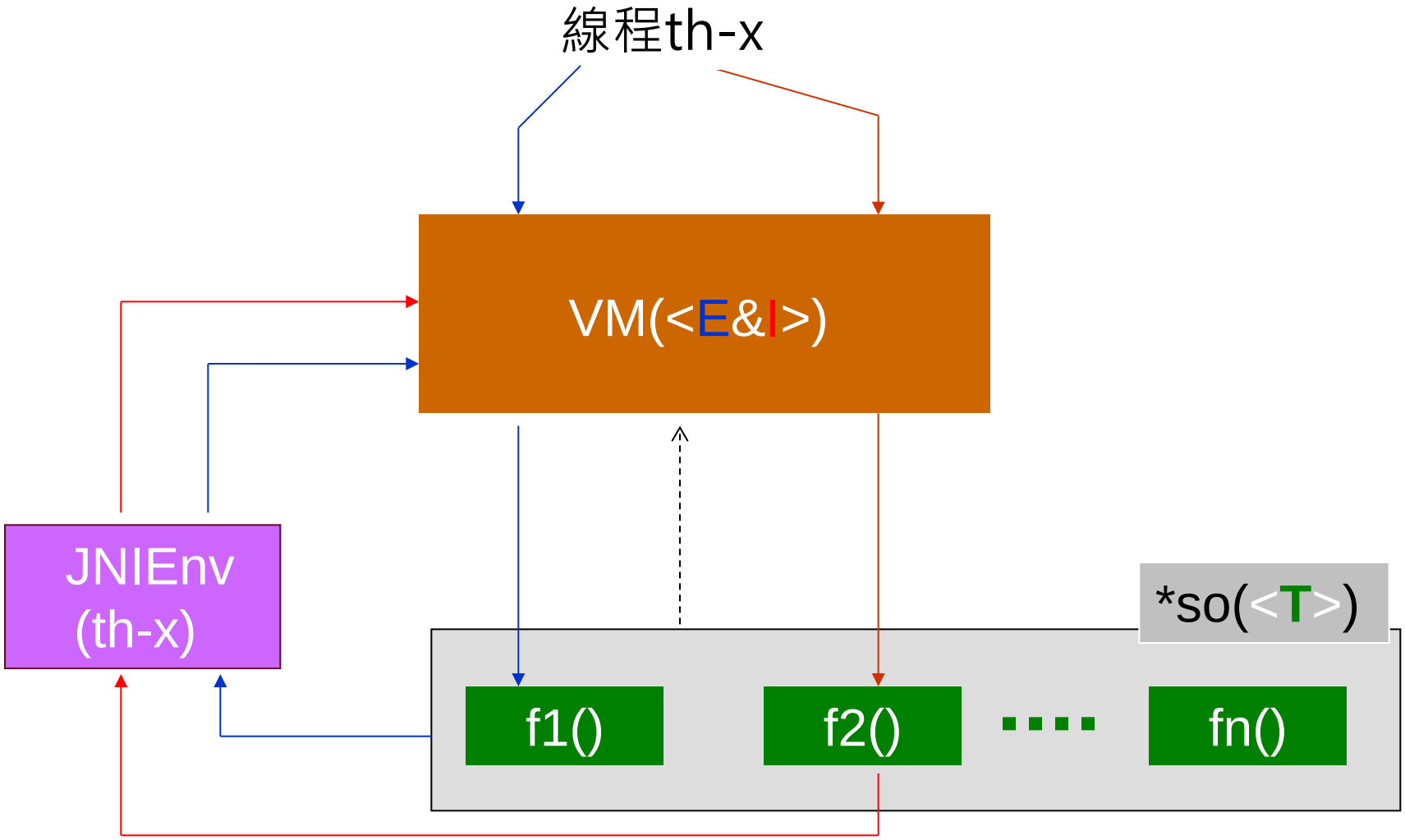
进程、线程和JNI架构(f)

By 高煥堂

問題

A3.4-JNI

- 接续上一个题目，基于JNI的EIT造形，让线程在执行插件<T>(就是*.so)时，都能随时透过JNIEnv对象来与<E&I>(就是 VM)来通信；如下图所示。
- 请问，在线程执行f1() 、f2()时，透过JNIEnv对象来与 VM来通信，有何目的呢?


線程th-x
VM(<E&I>)
JNIEnv
(th-x)
*so(<T>)
f1()
f2()
....
fn()

相关问题

- 请问：为什么<T>不直接与VM通信，而要透过JNIEnv对象呢?
- 请问：JNIEnv对象用来储存什么信息或数据呢?

問題

A3.5-JNI

- 接续上一个题目，不同的线程，会使用不同的JNIEnv对向来与 VM通信；如下述两个图所示。
- 请问，这样能有效化解多线程的冲突问题吗?

圖3.5A

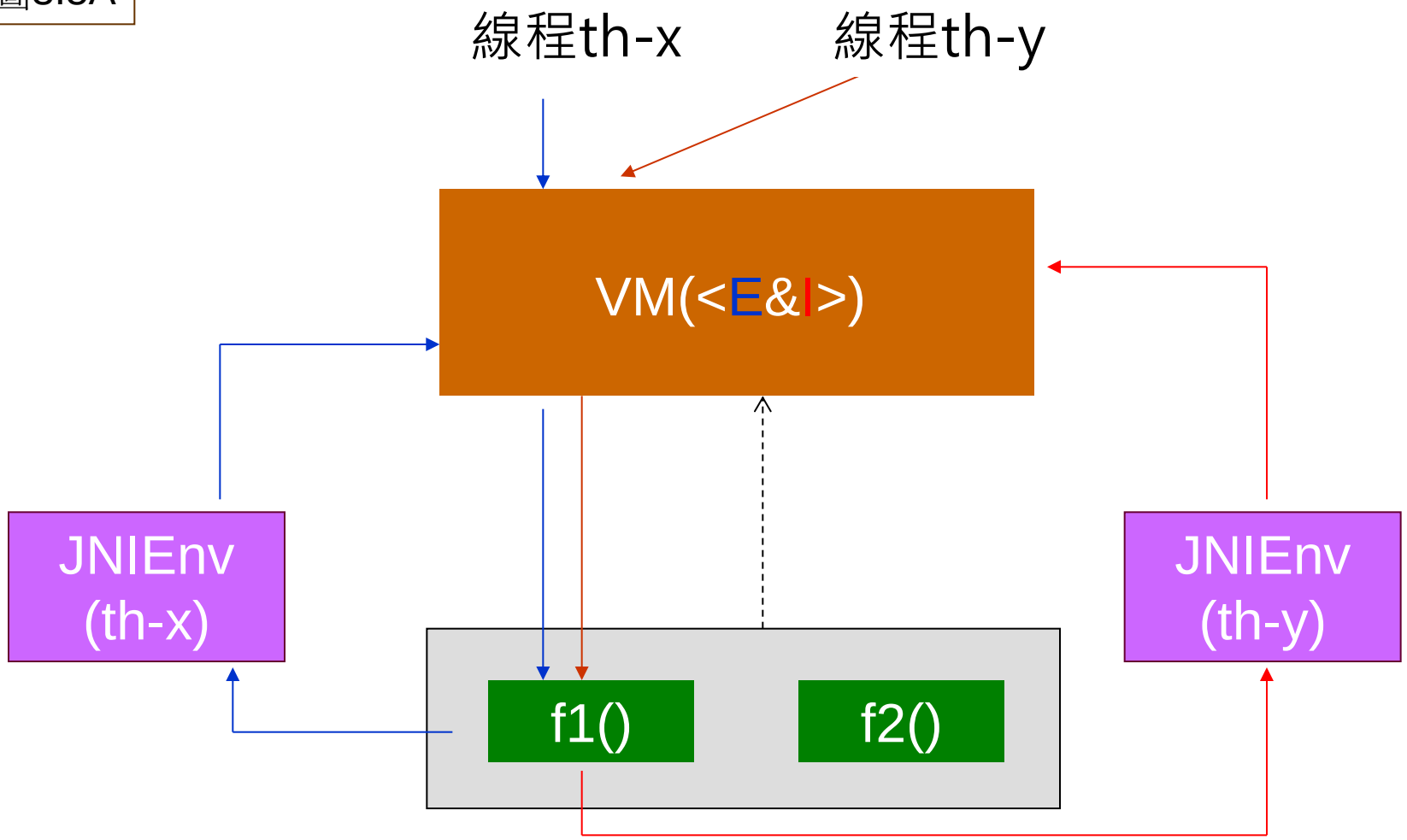
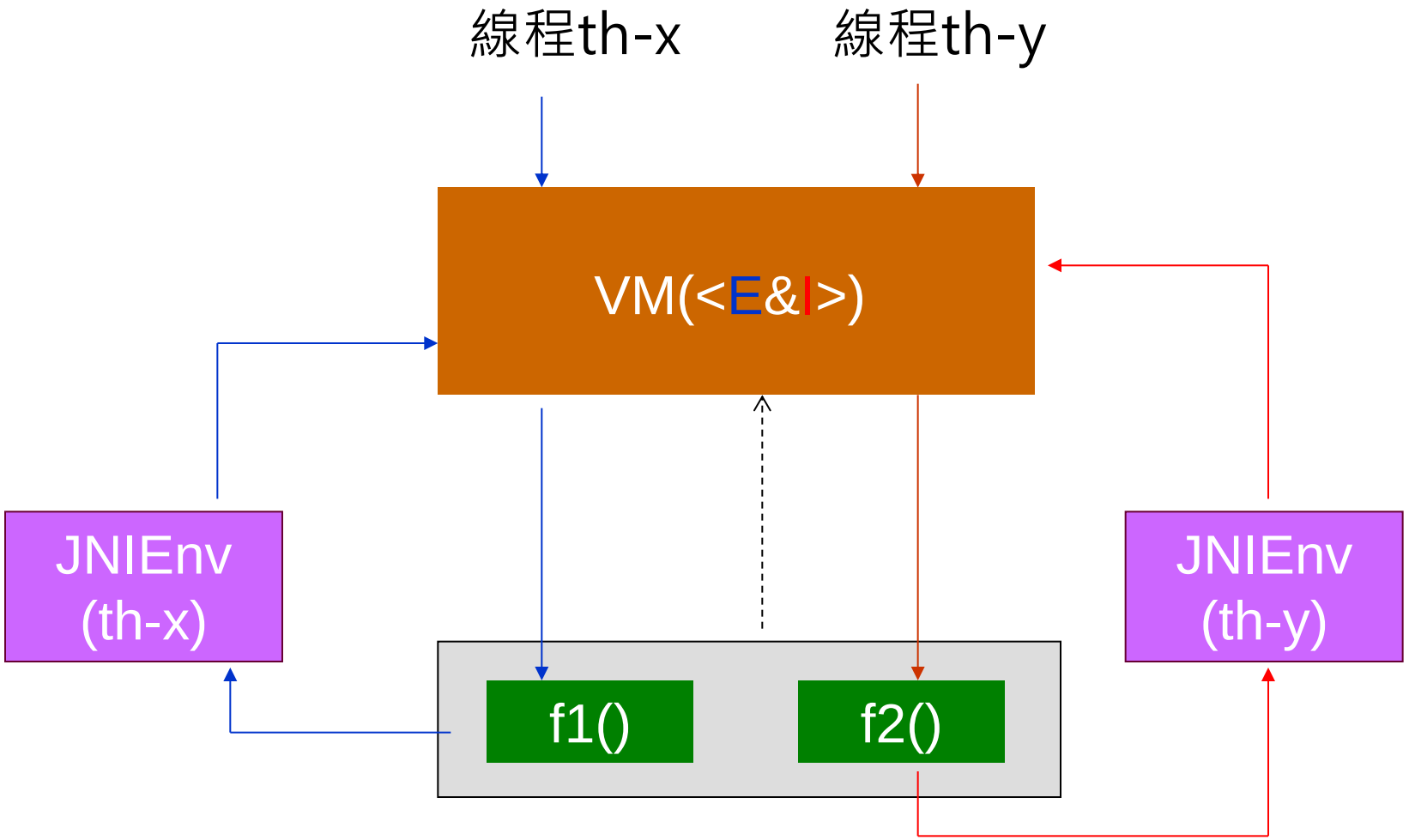


圖3.5B



提示

- 回想，当我们在写Web程序时，都会用到Session对象。
- 这个Session对象与JNIEnv对象的角色是一致的；两者有异曲同工之妙。
- 每次浏览器建立完成一个Connection(如同一条线程)，都会获得一个专用的Session对象。

問題

A3.6-JNI

- 接续上一个题目。有一个线程(th-x)先进入EIT造形去执行f1()，如下图3.6的Step-1所示。
- <T1>做了1+2+3+ … + 10的计算，并计算出结果(即sum值) 。然后，th-x就离开f1()返回到Java层了。
- 随后，线程(th-x)再度进入EIT造形去执行f2()插件，想取得先前f1()所计算出来的sum值。如下图3.6的Step-2所示。

- 请问：f1()插件应该将sum值储存在哪里，才能让f2()顺利拿到sum值呢?

圖3.6： Step-1

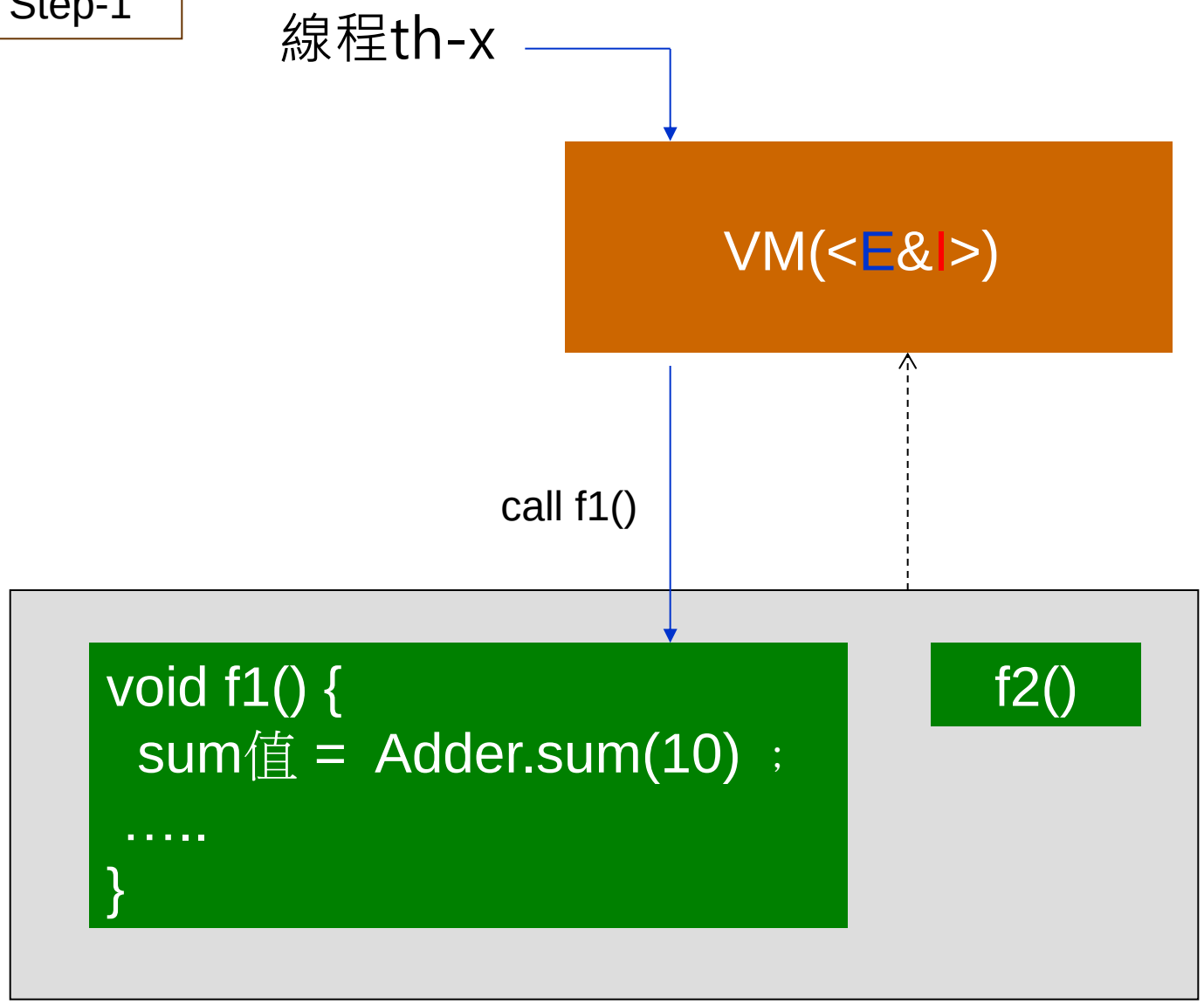
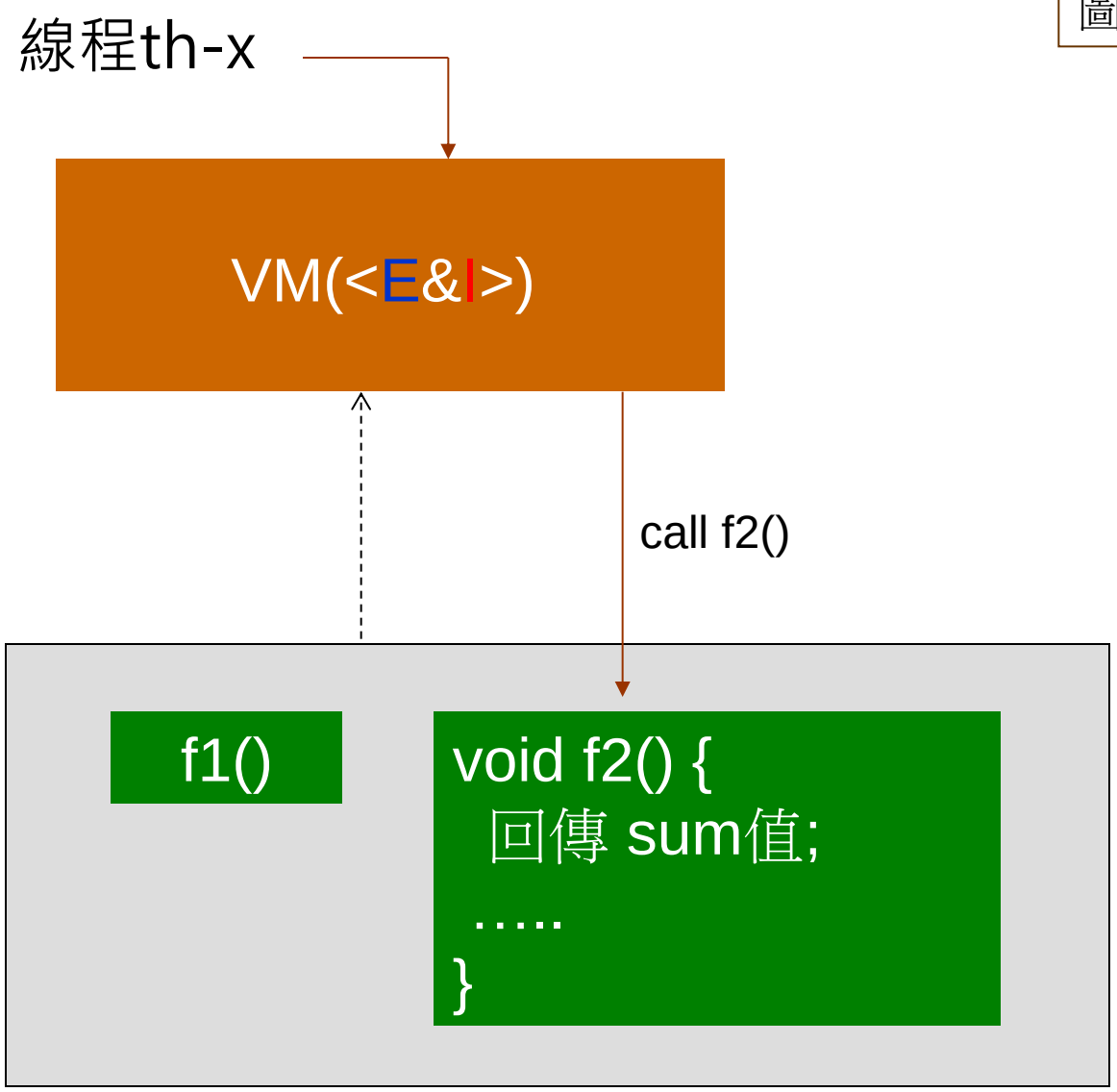


圖3.6： Step-2



提示

- 由于，同一条线程去执行f1()和f2()，此时f1()和f2()都能存取同一个JNIEnv对象。
- 能不能透过*.so的公用变量(Global Variable)呢?

問題

A3.7-JNI

- 接续上一个题目。刚才是同一个线程(th-x)去执行f1()和f2()插件。
- 如果分别由不同的线程去分别执行f1()和f2()插件的话，f2()想取得先前f1()所计算出来的sum值。如下图3.7所示。

圖3.7： Step-1

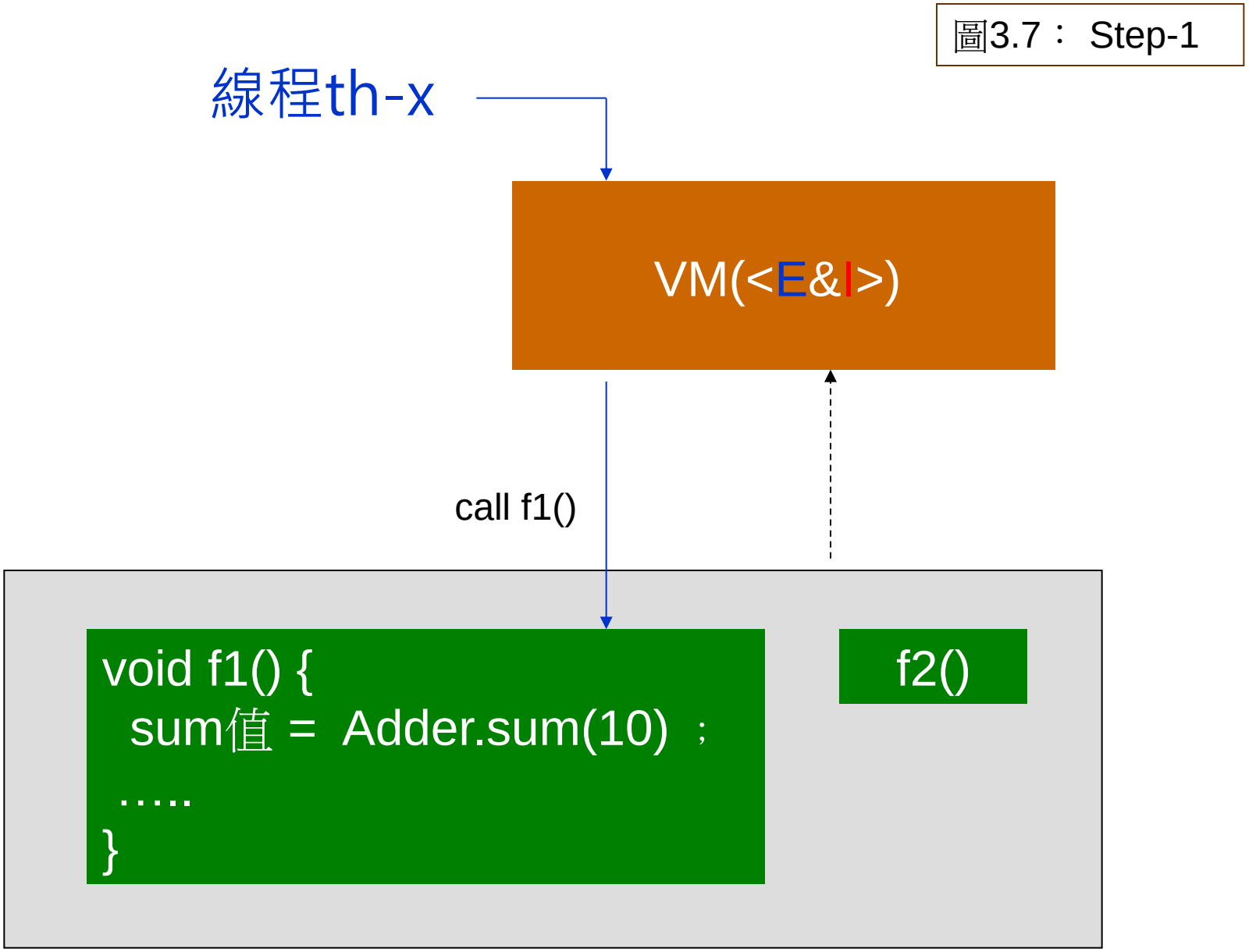
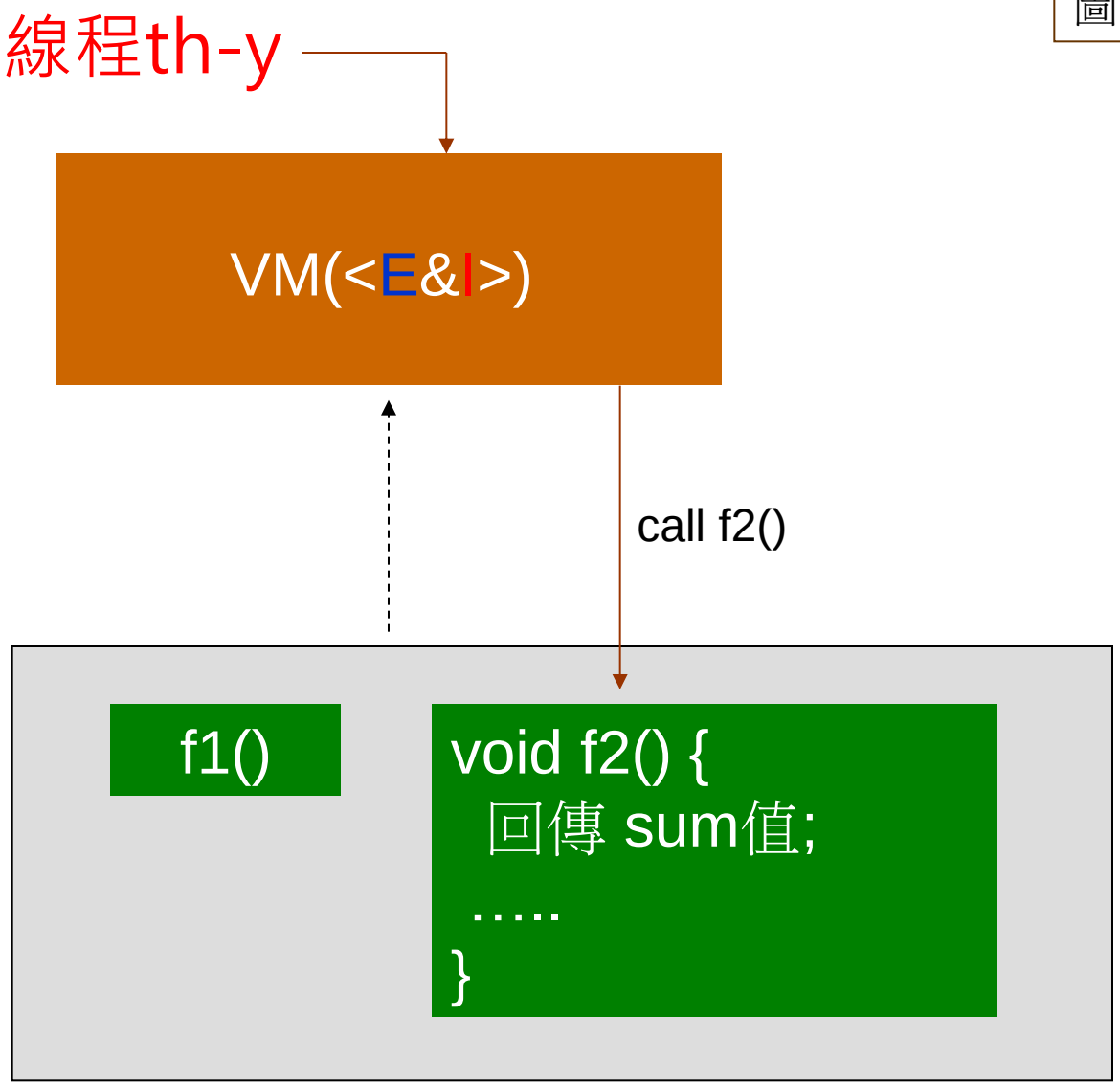


圖3.7： Step-2



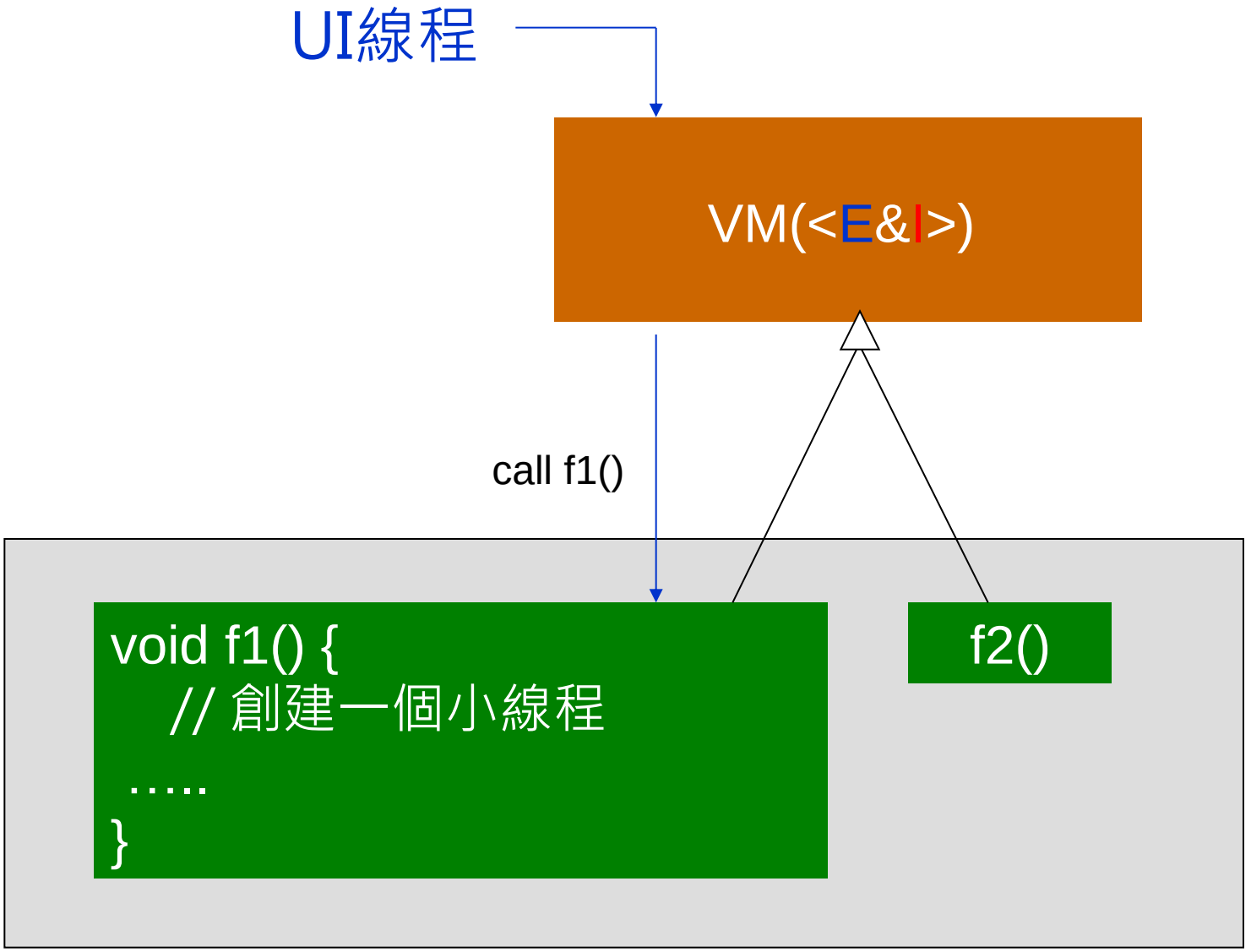
提示

- 由于，由不同的线程去执行f1()和f2()，此时f1()和f2()使用不同的JNIEnv对象。
- 能不能透过*.so的公用变量(Global Variable)呢?

問題

A3.8-JNI

- 当 UI线程经由VM而去执行<T>时，如下图所示。
- 请问：在执行<T>时，UI线程如何去创建一个小线程呢?
- 如何撰写其代码(创建一个小线程)呢?


UI線程
VM(<E&I>)
call f1()
void f1() {
// 創建一個小線程
.....
}
f2()

相关问题

- 在<T>里所创建的小线程，VM有给它专属的JNIEnv对象吗?
- 如果没有的话；请问：如何去向VM索取一个JNIEnv对象呢?
- 索取JNIEnv对象，有何目的呢?

```
/* com.misoo.counter.CounterNative.cpp */
#include <stdio.h>
#include <pthread.h>
#include "com_misoo_counter_CounterNative.h"
jmethodID mid;
jclass mClass;
JavaVM *jvm;
pthread_t thread;
int n, sum;
void* trRun( void* );

void JNICALL
Java_com_misoo_counter_CounterNative_nativeSetup(
        JNIEnv *env, jobject thiz) {
    jclass clazz = env->GetObjectClass(thiz);
    mClass = (jclass)env->NewGlobalRef(clazz);
    mid = env->GetStaticMethodID(mClass, "callback", "(I)V");
 }
```

```
void JNICALL
Java_com_misoo_counter_CounterNative_nativeExec
(JNIEnv *env, jobject thiz, jint numb){
    n = numb;
    pthread_create( &thread, NULL, trRun, NULL);
 }
void* trRun( void* ){
     int status;
    JNIEnv *env;    bool isAttached = false;
    status = jvm->GetEnv((void **) &env, JNI_VERSION_1_4);
    if(status < 0) {
        status = jvm->AttachCurrentThread(&env, NULL);
        if(status < 0)  return NULL;
        isAttached = true;
      }
    sum = 0;
    for(int i = 0; i<=n; i++)  sum += i;
    env->CallStaticVoidMethod(mClass, mid, sum);
    if(isAttached)  jvm->DetachCurrentThread();
    return NULL;
}
```

~ Continued ~