Software Engineering

WS 2022/23, Sheet 05

Prof. Dr. Sven Apel
Yannick Lehmen, Maurice Vincon

Handout: 28.11.2022

Task 1

a) How can we implement variability using build systems? How fine grained can feature specific regions in the codebase be compared to other mechanisms like preprocessors?

b) **Discussion:** Prepare arguments for and against implementing variability by using build systems. You can present your arguments during the discussion in the tutorial sessions.

Solution

a)
- one build script per variant
- include or exclude files before compilation
- variant–specific files replace base files
- granularity at the file–level as opposed to preprocessors (code–level)

Task 2

a) **Discussion:** Prepare arguments for and against implementing variability by using preprocessors. You can present your arguments during the discussion in the tutorial sessions.

b) Find the configurations that could cause errors in the following piece of code. What are these errors?.

```
1  a = 1; b = 1;
2  #ifdef X
3  a++;
4  #ifdef Y
5  a = a * 2;
6  #endif
7  #endif
8  #ifdef Z
9  b = 4;
10 #ifdef X
11 a = a - 2;
12 #elif W
13 b = b - 1;
14 #else
15 b = 5;
16 #endif
17 b = b / a;
18 #endif
```

Solution

b) Error(s):

```
1  a = 1; b = 1;
2  #ifdef X
3  a++;
4  #ifdef Y
5  a = a * 2;
6  #endif
```

```
#endif
#ifdef Z
b = 4;
#ifdef X
a = a - 2;
#elif W
b = b - 1;
#else
b = 5;
#endif
b = b / a;  // Division by 0: X ∧ ¬Y ∧ Z
#endif
```

Task 3

a) **Discussion:** Prepare arguments for and against implementing variability by using version control systems. You can present your arguments during the discussion in the tutorial sessions.

b) Describe the differences between variants, revisions, and versions.

c) Describe the differences between structured and unstructured merge.

d) Find the merge conflicts and errors in the following example, where you merge the modules *Main Change 1* and *Main Change 2* as separate changes of the main class, for:

(i) unstructured merge

(ii) structured merge

```java
public class Main {

  public static boolean featureA = true;

  public static void main(String[] args) {
    int a = 25;
    int b = 5;

    if (featureA) {
      System.out.println(
          "Feature A is activated.");
      b = a / b;
      System.out.println(
          "Feature value is: "+b);
    } else {
      System.out.println(
          "Feature A is deactivated.");
    }
  }
}
```

```java
//Main Change 1
public class Main {

  public static boolean featureB = true;

  public static void main(String[] args) {
    int a = 0;
    int b = 25;

    if (featureB) {
      System.out.println(
          "Feature B is activated.");
      b = a * b;
      System.out.println(
          "Feature value is: "+b);
    } else {
      System.out.println(
          "Feature B is deactivated.");
    }
  }
}
```

```java
//Main Change 2
public class Main {

  public static boolean featureA = true;

  public static void main(String[] args) {
    int a = 25;
    int b = 5;

    if (featureA) {
      System.out.print(
          "Feature A is activated.");
      b = a / b;
      boolean c = featureA;
      int d = b / a;
      System.out.println(
          "Feature value is: "+b+"&"+d);
    } else {
      System.out.println(
          "Feature A is deactivated.");
    }
  }
}
```

Solution

b) A variant is a configured version of a software project, e.g., a graph library with support for weighted edges. Revisions comprise all the development stages of a software project. Versions are all variants and revisions.

c) An unstructured merge takes two changes of a module and merges them together line by line. If there are changes in the same line in both modules, the merge runs into a conflict that has to be resolved manually. A structured merge takes the structure of the modules into account (i.e., their ASTs). This additional information makes it possible to avoid certain conflicts that the unstructured merge cannot resolve.

d)
- unstructured

```
1  //Main Merge unstructured
2  public class Main {
3
4    public static boolean featureB = true;
5
6    public static void main(String[] args) {
7      int a = 0;
8      int b = 25;
9
10     if (featureB) {
11 <<<<<<< Main-2.java
12       System.out.println(
13           "Feature B is activated.");
14 =======
15       System.out.print(
16           "Feature A is activated.");
17 >>>>>>> Main-3.java
18       b = a * b;
19       boolean c = featureA;
20       int d = b / a;
21       System.out.println(
22           "Feature value is: "+b+"&"+d);
23     } else {
24       System.out.println(
25           "Feature B is deactivated.");
26     }
27   }
28 }
```

- structured

```java
//Main Merge Structured
public class Main {

  public static boolean featureB = true;

  public static void main(String[] args) {
    int a = 0;
    int b = 25;

    if (featureB) {
      System.out.print(
          "Feature B is activated.");
      b = a * b;
      boolean c = featureA; //featureA not present!
      int d = b / a; //division by 0!
      System.out.println(
          "Feature value is: "+b+"&"+d);
    } else {
      System.out.println(
          "Feature B is deactivated.");
    }
  }
}
```