Intention-based variant integration

Wilhelm Hedman
Chalmers University of Technology

In collaboration with: Max Lillack, Stefan Stanciulescu, Thorsten Berger, Andrzej Wasowski

Slides adapted from Stefan!
Agenda

- Background: Clones, clones, clones
- Solution: Intention-based clone integration
- Evaluation: Replaying integration scenarios
Clones: quick and easy with high maintenance costs
Re-engineering to enable systematic reuse

Main

Cloned Variant1

Cloned Variant2

Derive the cloned variants

Patch

Derive the cloned variants

Patch

Patch
Variant integration vs. regular merging

- **Variant integration**: cohabilitating features to enable variants (semantics!)
- **Regular merging**: greedy - delegate conflicts (syntax!)
- Goals are different:
  - Ensuring that multiple features work together, contra
  - Single implementation of parallel changes
Diffing to the rescue?

```c
#ifdef ULTIPANEL
    ...
    uint8_t lastEncoderBits;
    uint32_t encoderPosition;
    #if PIN_EXISTS(SD_DETECT)
        uint8_t lcd_sd_status;
    #endif
#endif // ULTIPANEL
menu_t cm = lcd_status_scrn;
bool ignore_click = false;
```
Diffing to the rescue?

```c
#define ULTIPANEL
uint8_t lastEncoderBits;
uint32_t encoderPosition;
#if PIN_EXISTS(SD_DETECT)
uint8_t lcd_sd_status;
#endif
#endif // ULTIPANEL

menu_t cM = lcd_status_scrn;
bool ignore_click = false;
```

Mainline

```c
#define ULTIPANEL
uint8_t lastEncoderBits;
uint32_t encoderPosition;
#if (SDCARDDETECT > 0)
int8_t encoderDiff;
uiint32_t encoderPosition;
bool lcd_oldcardstatus;
#endif
#endif // ULTIPANEL

menu_t cM = lcd_status_scrn;
```

Fork
Diff doesn’t work :(

```diff
...  
#ifdef FORK
    #if (SDCARDDETECT > 0)
        bool lcd_oldcardstatus;
    #else
        #if PIN_EXISTS(SD_DETECT)
            uint8_t lcd_sd_status;
    #endif
    #endif
...  
```
Proper integrated AST is hard to construct by hand

```c
#define ULTIMAPANEL
uint8_t lastEncoderBits;
uint32_t encoderPosition;
#else PIN_EXISTS(SD_DETECT)
uint8_t lcd_sd_status;
#endif // ULTIMAPANEL

#define ULTIMAPANEL
uint8_t lastEncoderBits;
uint8_t encoderDiff;
uint32_t encoderPosition;
#else FORK
#endif
#elif SDCARD_DETECT > 0
bool lcd_oldcardstatus;
#endif
#else
#endif
#endif
#endif
#endif
#endif
#endif
```
Our goal:

Support *re-engineering* of clone-based *variants* into software product lines using *intentions* and *views*

Achieved by:

- Abstraction from source code
- Intuitive intentions
- Views to explore results
- Interactive process
Integration process:

1. Automatically generate integrated AST from two variants
2. Explore integrated AST using views
3. Edit variational AST - add integration intentions
Integration process:

1. Automatically generate integrated AST from two variants
2. Explore integrated AST using views
3. Edit variational AST - add integration intentions

Benefits:
- The code can be compiled
- Test suites can be run
- Variants can be derived
Views and intentions in action
What are integration intentions?

- Intentions are intuitive declarations reflecting the developer’s integration goal
  - e.g., keep functionality, remove functionality, keep as configurable feature
- Declared on blocks of code, shown in the different views
- Control the desired structure of the integrated file
- Intentions are automatically resolved on the integrated AST

**Benefits:** raise abstraction level from \#ifdef structures to intuitive intentions
Intentions:

- Keep
- Remove
- KeepAsFeature
- Exclusive
- AssignFeature
- Postpone
Intentions: Keep

- Keep
- Remove
- KeepAsFeature
- Exclusive
- AssignFeature
- Postpone

```
#ifndef FORK // block_not_fork
    int servo_e1[] = SE
    int servo_e2[] = SEA
#endif // block_fork

#else // block_fork
    int16_t servo_e1 = SE
    int16_t servo_e2[] = SEA
#endif
```

Keep intention

Result

int servo_e1[] = SE
int servo_e2[] = SEA
#ifndef FORK
    int16_t servo_e1 = SE
    int16_t servo_e2 = SEA
#endif
Intentions: Exclusive

- Keep
- Remove
- KeepAsFeature
- Exclusive
- AssignFeature
- Postpone

```
#ifndef FORK //block2
    lcd.print(msg);
#else
    #ifdef FIL_DISPLAY //block1
        if(condition){
            lcd.print(msg);
        }
    
#else
    lcd.print(trnsf(data));
#endif
#endif
```

```
#ifndef FIL_DISPLAY
    lcd.print(msg);
#else
    if(condition){
        lcd.print(msg);
    } else{
        lcd.print(trnsf(data));
    }
#endif
```

*Exclusive* intention  

Result
Evaluation - so far

1. **Completeness** - intentions suffice
2. **Correctness** - intentions execution produces correct results
3. **Efficiency** - using intentions is faster than using unstructured approach

Method: Replay merge commits using ordinary tool and prototype tool.

1. Well, do they?
2. Check that output is well-formed.
3. Record number of edit operations.
Evaluation observations (so far)

1. **Completeness:** The intentions suffice for performing common integration tasks. Often, just using *Keep* and *Remove* resolve the task.

2. **Correctness:** When the intentions are correctly declared, they produce a correctly integrated configurable platform.

3. **Efficiency:** Developers need to perform substantially fewer operations using our approach.
Evaluation - next step

Challenges:

● Getting more examples (open source + industry projects)
● Controlled experiment, given better tool and intentions:
  ○ User study: students and professional developers to perform integration tasks
  ○ Compare time/efficiency and correctness
Summary

Diff doesn't work :(  
...
#define FORK
#define DEBUG
#endif

diff -D FORK

Fixed missing #endif from merge

In action!

Intentions: Keep
- Keep
- Remove
- KeepAsFeature
- Exclusive
- AssignFeature
- Postpone

Evaluation - so far
1. Completeness - intentions suffice
2. Correctness - intentions execution produces correct results
3. Efficiency - using intentions is faster than using unstructured approach

Method: Replay merge commits using ordinary tool and prototype tool.
1. Well, do they?
2. Check that output is well-formed.
3. Record number of edit operations.