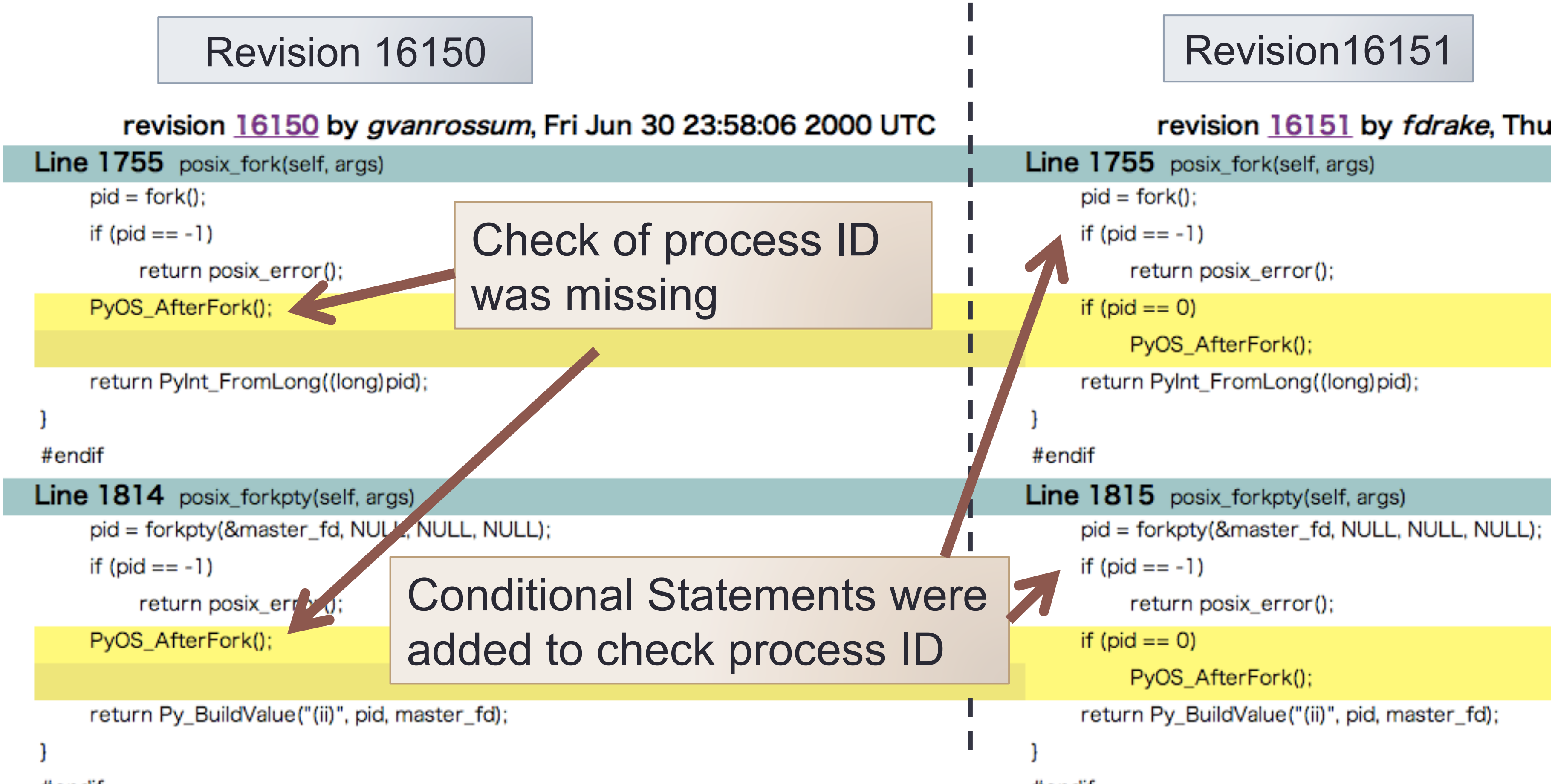
HOW WE KNOW THE PRACTICAL IMPACT OF CLONE ANALYSIS

Norihiro Yoshida (Nara Institute of Science and Technology)
Eunjong Choi (Osaka University)
Yuki Yamanaka (Osaka University)
Katsuro Inoue (Osaka University)

Simultaneous editing of clones in the Python project

Function-level clones for *fork* model variations in UNIX systems.



Revision 16150

revision 16150 by *gvanrossum*, Fri Jun 30 23:58:06 2000 UTC

Line 1755 posix_fork(self, args)

```
    pid = fork();
    if (pid == -1)
        return posix_error();
    PyOS_AfterFork();

    return PyInt_FromLong((long)pid);
}
#endif
```

Line 1814 posix_forkpty(self, args)

```
    pid = forkpty(&master_fd, NULL, NULL, NULL);
    if (pid == -1)
        return posix_error();
    PyOS_AfterFork();

    return Py_BuildValue("(ii)", pid, master_fd);
}
#endif
```

Revision16151

revision 16151 by *fdrake*, Thu

Line 1755 posix_fork(self, args)

```
    pid = fork();
    if (pid == -1)
        return posix_error();
    if (pid == 0)
        PyOS_AfterFork();
    return PyInt_FromLong((long)pid);
}
#endif
```

Line 1815 posix_forkpty(self, args)

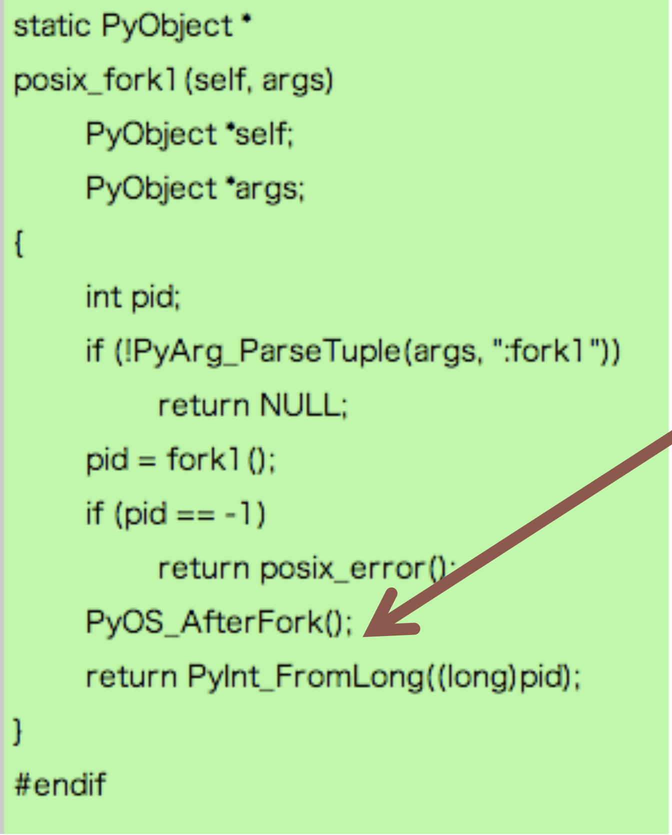
```
    pid = forkpty(&master_fd, NULL, NULL, NULL);
    if (pid == -1)
        return posix_error();
    if (pid == 0)
        PyOS_AfterFork();
    return Py_BuildValue("(ii)", pid, master_fd);
}
#endif
```

Check of process ID was missing

Conditional Statements were added to check process ID

Same bug was introduced again

- At a later date, a function-level clone was added for the another variation of fork model.



```
static PyObject *
posix_fork1(self, args)
        PyObject *self;
        PyObject *args;
{
        int pid;
        if (!PyArg_ParseTuple(args, ":fork1"))
                return NULL;
        pid = fork1();
        if (pid == -1)
                return posix_error();
        PyOS_AfterFork();
        return PyInt_FromLong((long)pid);
}
#endif
```

Check of process ID was missing again!

This bug was also fixed in a later revision.

This is a good motivated example for clone analysis but …

We do not know the practical impact of clone analysis.

Q1: Did the developers use clone analysis tools to perform the simultaneous editing and the bug fix?

Q2: Can clone analysis tools successfully support the developers once they use?

For the advancement of clone analysis tools,
we should observe how those techniques affect development.

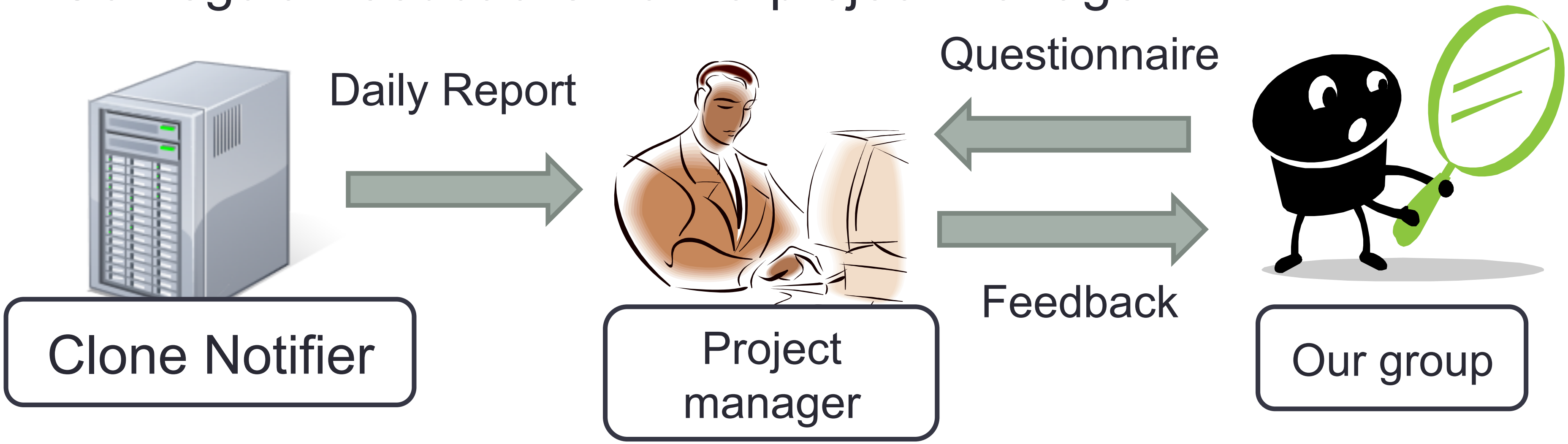
How should we observe the impact of applying clone analysis techniques during actual development process?

Our experience of observing the practical impact of clone analysis

- Applied Clone Notifier to development process in NEC
  - 6 programmers, 120 KLOC written in Java

- Clone Notifier: Clone change notification system for notifying change of code clones.

- Got regular feedbacks from a project manager



Daily Report

Questionnaire

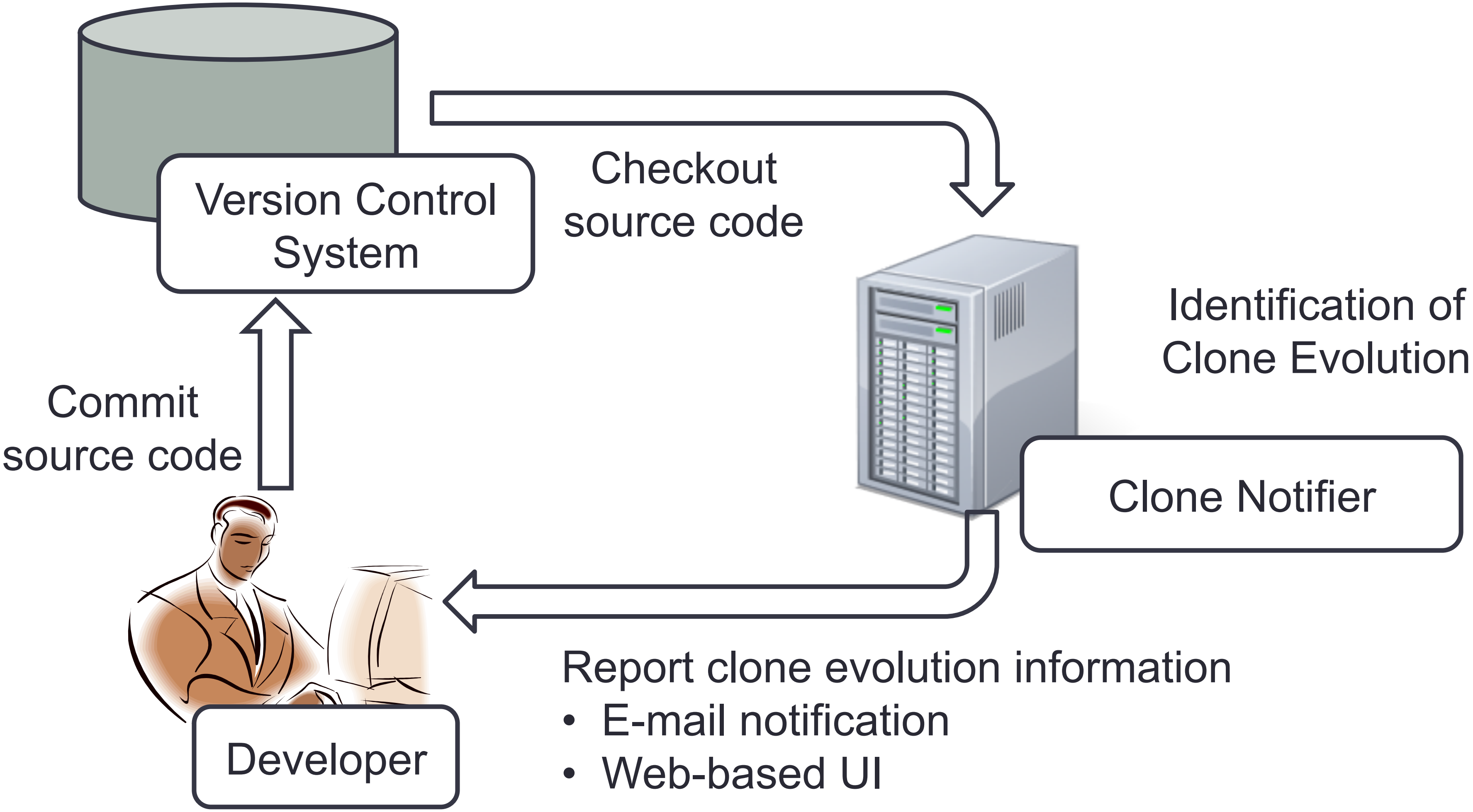
Feedback

Clone Notifier

Project manager

Our group

Clone Notifier :
A Clone Change Notification System [1]



Version Control System

Checkout source code

Identification of Clone Evolution

Clone Notifier

Commit source code

Developer

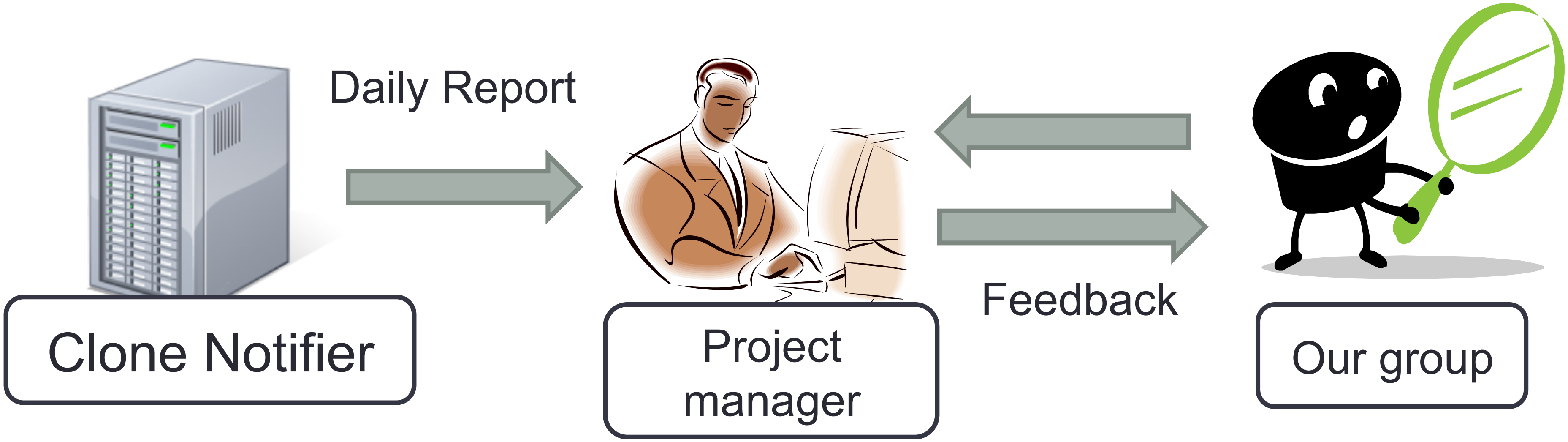
Report clone evolution information

- E-mail notification
- Web-based UI

[1] Yamanaka, et al.: "Applying Clone Change Notification System into an Industrial Development Process", ICPC 2013.

In order to get feedbacks

- We asked the project manager whether he found unintentionally-developed clones using Clone Notifier.
- If he said yes, we asked him how the development team should maintain them.
  - Perform refactoring
  - Write a source code comment to denote the existence of the clone
  - leave the clone as it is



Feedbacks from the project manager

He recognized

- 10 clone sets should be refactored
  - Two of the 10 clone sets was merged into a single function during the 40 days, respectively.
  - The other 8 clone sets were designated as refactoring candidates for next maintenance project.

- 1 clone set should be noted in a source code comment

Findings from Manual Observation

- All refactoring opportunities were newly-appeared by adding new code.

  → If code clones are newly-appeared by only code replacement and deletion, they should not be notified.

- All refactoring opportunities were syntactically-complete.
  - E.g., whole parts of loop or branch statements are involved.

  → Syntactically-incomplete clones should not be notified.

Summary & Challenge

We introduce our experience in observing the practical impact of clone analysis in industry.

We hope that our work provides a good starting point to discuss approaches for observation.

Challenge:

- More disciplined approaches for observation
- Larger-scale observation
- Approaches to sharing observation results among the clone research community