Typical debugging process

Debugging is a methodical process of finding and reducing the number of bugs or defects in a software program, or a piece of computer hardware. Thus, it is a result of behavior as expected. Debugging tends to be harder when various subsystems are interdependent, especially when changes in one may cause bugs to emerge in another. Many books have been written about debugging (see below: Further reading). It involves numerous aspects, including interactive debugging, static analysis, source level monitoring, binary debugging, program visualization, program cloning, and special design tactics to improve detection while simplifying changes.

Contents
1 Origin
2 Process
3 Tools
4 Typical debugging process
5 Techniques
6 Debugging for embedded systems
7 Anti-debugging
8 See also
9 References
10 Further reading
11 External links

Origin
There is some controversy over the origin of the term "debugging".

The terms "bug" and "debugging" are both popularly attributed to Grace Hopper at Harvard University. In 1945, while she was working on a Mark II Computer at Harvard University, her associates discovered a moth stuck in a relay and thereby impeding operation, whereupon she remarked that they were "debugging" the system. However, the term "bug" (in the meaning of technical errors) dates back at least to 1838 and Thomas Edison used a "light bulb bug" for a full discussion, and "debugging" seems to have been used as a term in aerodynamics before entering the world of computers. Indeed, in an interview Grace Hopper remarked that she was not coining the term. The moth filled the already existing terminology, so it was saved.

The Oxford English Dictionary, entry for "debug" quotes the term "debugging" used in reference to airplane engine testing in a 1945 article by C. B. Ogilvie. An article in The Royal Aeronautical Society, Hopper's bug was found on September 9, 1947. The term was not adopted by computer programmers until the early 1950s. The seminal article by Gill in 1963 is the earliest in-depth discussion of programs and bugs, but it does not use the term "bug" or "debugging." In the ACM digital library, the term "debugging" is first used in three papers from 1962 ACM National Meetings.

Supporting disciplines
- Configuration management
- Software quality assurance (SQA)
- Software maintenance
- Tools
  - Debuggers
  - Compressor Debugger (PL/I debugger)
  - IDE build automation

Typical debugging process

Debugging is usually done with special hardware such as the Xbox debug unit intended only for developers.

Debugging ranges, in complexity, from fixing simple errors to performing lengthy and time-consuming tasks of data collection, analysis, and scheduling updates. The debugging skill of the programmer can be a major factor in the ability to debug a problem, but the difficulty of software debugging varies greatly with the complexity of the system, and also depends, to some extent, on the programming language(s) used and the available tools, such as debugging.

Debuggers are software tools which enable the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

Generally, high-level programming languages, such as Java, make debugging easier, because they have features such as exception handling, which take care of errors in the behavioral aspect of the system. In programming languages such as C or assembly, huge rely on the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

In certain situations, general-purpose software tools that are language specific in nature can be very useful. These take the form of Ada code analyzers only. These tools look for a very specific set of known problems, some common and some rare, within the Ada language. Debuggers are software tools which enable the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

As software and electronic systems have become generally more complex, the need to develop and debug programs has increased. Debugging is usually done with special hardware such as the Xbox debug unit intended only for developers.

Debugging ranges, in complexity, from fixing simple errors to performing lengthy and time-consuming tasks of data collection, analysis, and scheduling updates. The debugging skill of the programmer can be a major factor in the ability to debug a problem, but the difficulty of software debugging varies greatly with the complexity of the system, and also depends, to some extent, on the programming language(s) used and the available tools, such as debugging.

Debuggers are software tools which enable the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

Generally, high-level programming languages, such as Java, make debugging easier, because they have features such as exception handling, which take care of errors in the behavioral aspect of the system. In programming languages such as C or assembly, huge rely on the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

In certain situations, general-purpose software tools that are language specific in nature can be very useful. These take the form of Ada code analyzers only. These tools look for a very specific set of known problems, some common and some rare, within the Ada language. Debuggers are software tools which enable the programmer to monitor the execution of a program, step by step, set breakpoints, change values in memory. The term debugger can also refer to the person who is doing the debugging.

As software and electronic systems have become generally more complex, the need to develop and debug programs has increased. Debugging is usually done with special hardware such as the Xbox debug unit intended only for developers.
Debugging for embedded systems

In contrast to the general purpose computer software design environment, a primary characteristic of embedded environments is the sheer number of different platforms available to developers (CPU architectures, vendors, operating systems and their variants). Embedded systems are, by definition, not general-purpose designs: they are typically developed for a single task (or small range of tasks), and the platform is chosen specifically to optimize that application. Not only does this fact make life tough for embedded system developers, it also makes debugging and testing of these systems harder as well, since different debugging tools are needed in different platforms.

To identify and fix bugs in the system (e.g. logical or synchronization problems in the code, or a design error in the hardware);
to collect information about the operating states of the system that may then be used to analyze the system: to find ways to boost its performance or to optimize other important characteristics (e.g. energy consumption, reliability, real-time response etc.).

Anti-debugging

Anti-debugging is “the implementation of one or more techniques within computer code that hinders attempts at debugging a target process”. Anti-debugging is “the implementation of one or more techniques within computer code that hinders attempts at debugging a target process”.

or debugging a target process. Anti-debugging is “the implementation of one or more techniques within computer code that hinders attempts at debugging a target process”.

Techniques

Print (or tracing) debugging is the act of watching (live or recorded) trace statements, or print statements, that indicate the flow of execution of a process. This is sometimes called print debugging, due to the use of the printf statement in C.

Remote debugging is the process of debugging a program running on a system different than the debugger. To start remote debugging, debugger connects to a remote system over a network. Once connected, debugger can control the execution of the program on the remote system and retrieve information about it’s state.

Post-mortem debugging is debugging of the program after it has already crashed. Related techniques often include various tracing techniques (for example and/or analysis of core dumps or memory dumps) of the crashed process. The dump of the process could be obtained automatically by the system (for example, when process has terminated due to an unexpected exception), or by a programmer-reassembled instruction, or manually by the interactive user.

 Delta Debugging - technique of automating test case simplification

Saftey Squaress - technique of isolating failure within the test using progressive inlining of parts of the failing test.

Software testing

Software testing is the act of finding programming errors, vulnerabilities, or other flaws in a computer program. The testing process includes the selection of test data, execution of the tests, and inspection of the results. Testing is a means of verifying that a program satisfies its specifications or meets the requirements.

References

* G. Hopper from FLODOC
* Kent Beck, Test Driven: Refactoring and the Safety Squaress,
* Software Protection through Anti-Debugging Viewers, Stephen Taylor.

Further reading


External links

* Crash dump analysis patterns
* In-depth articles on analyzing and finding bugs in crash dumps
* Embedded Systems test and debug - about digital input generator
* Results of a survey about embedded system test and debug
* Byte Paradigm

From Wikipedia, the free encyclopedia
<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sebelumnya</td>
<td>Methodologies Waterfall Prototype model Incremental Iterative V Model Spiral Scrum Cleanroom RAD DSDM RUP XP Agile Lean Dual Vee</td>
</tr>
<tr>
<td>Iklan</td>
<td>Activities and steps Requirements Specification Architecture Construction Design Testing Debugging Deployment Maintenance</td>
</tr>
<tr>
<td>Klik disini untuk Kembali ke-sebelumnya</td>
<td>(Debugger)</td>
</tr>
<tr>
<td>Tutorial, Referensi, Buku, Petunjuk</td>
<td>wisata.us</td>
</tr>
<tr>
<td>• Koran &amp; Majalah Indonesia</td>
<td>• Hosting Indonesia</td>
</tr>
</tbody>
</table>

### Klik disini untuk Kembali ke-sebelumnya

- Sebelumnya
- Iklan
- Tutorial, Referensi, Buku, Petunjuk
- • Koran & Majalah Indonesia
- • Hosting Indonesia

### PHP

| Jakarta Timur
| --- |
| Jakarta Barat
| --- |
| Jakarta Pusat
| --- |
| Jakarta Selatan
| --- |
| Jakarta Timur
| --- |

### Berikutnya

- Jakarta Timur
- Jakarta Barat
- Jakarta Pusat
- Jakarta Selatan
- Jakarta Timur
debugging, debugging javascript

Debugging is the process of finding and resolving of defects that prevent correct operation of computer software or a system. A typical debugging process will normally the first step in debugging is to attempt to reproduce the problem. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Specific user environment and usage history can make it difficult to reproduce the problem.