EUROPEAN PATENT OFFICE
U.S. PATENT AND TRADEMARK OFFICE

CPC NOTICE OF CHANGES 1769

DATE: AUGUST 1, 2025

PROJECT DP12724

The following classification changes will be effected by this Notice of Changes:

Action	**Subclass**	**Group(s)**
DEFINITIONS:		
Definitions Modified:	G06F	11/00, 11/36, 11/3604, 11/362, 11/3668, 11/3698

No other subclasses/groups are impacted by this Notice of Changes.

This Notice of Changes includes the following *[Check the ones included]:*

1. CLASSIFICATION SCHEME CHANGES
   - ☐ A. New, Modified or Deleted Group(s)
   - ☐ B. New, Modified or Deleted Warning(s)
   - ☐ C. New, Modified or Deleted Note(s)
   - ☐ D. New, Modified or Deleted Guidance Heading(s)
2. DEFINITIONS
   - ☒ A. New or Modified Definitions (Full definition template)
   - ☐ B. Modified or Deleted Definitions (Definitions Quick Fix)
3. ☐ REVISION CONCORDANCE LIST (RCL)
4. ☐ CHANGES TO THE CPC-TO-IPC CONCORDANCE LIST (CICL)
5. ☐ CHANGES TO THE CROSS-REFERENCE LIST (CRL)

2. A. DEFINITIONS (modified)

G06F 11/00

Definition statement

Replace: The existing Definition statement with the following updated statement.

- Error avoidance (G06F11/004);
- Identification related to error detection/correction or monitoring (G06F11/006);
- Reliability and availability analysis of computing systems (G06F11/008);
- Error detection and/or correction (G06F11/07);
- Detection or location of defective computer hardware by testing at a time outside of "normal operating mode", e.g. during standby, idle time or at power on (G06F11/22);
- Checking the correct order of processing (G06F11/28);
- Monitoring on computing systems (G06F11/30);
- Prevention of errors by analysing, debugging or testing of software (G06F11/36).

References

Limiting references

Replace: The existing Limiting references table with the following updated table.

Error detection, correction or testing in information storage based on relative movement between record carrier and transducer	G11B20/18
Monitoring, i.e. supervising the progress of recording or reproducing	G11B27/36
Checking stores for correct operation; Testing stores during standby or offline operation	G11C29/00

Informative references

Replace: The existing Informative references table with the following updated table.

Testing of electronic circuits	G01R31/28
Testing or monitoring of control systems or parts thereof	G05B23/00

Methods or arrangements for verifying the correctness of marking on a record carrier	G06K5/00
Monitoring patterns of pulse trains	H03K5/19
Coding, decoding or code conversion, for error detection or error correction; Coding theory basic assumptions; Coding bounds; Error probability evaluation methods; Channel models; Simulation or testing of codes	H03M13/00
Arrangements for detecting or preventing errors in the information received	H04L1/00
Arrangements for monitoring or testing data switching networks	H04L43/00
Network arrangements, protocols or services for recovering from a failure of a protocol instance or entity	H04L69/40

Special rules of classification

Replace: The existing Special rules text with the following updated text.

The error detection or correction process in neural networks is also covered (G06F11/1476).

Glossary of terms

Replace: The existing Glossary of terms table with the following updated table. Please note that in addition to the lowercase edits in both columns, there are other minor edits throughout, so it should all be replaced.

fault	physical defect, imperfection or flaw that occurs within some hardware component, or logical defect of a piece of software. Essentially, the definition of a fault, as used in the fault tolerance community, agrees with the definition found in the dictionary. Faults may be permanent, transient or intermittent.
error	the logical manifestation of a fault, observable in terms of incorrect instructions of or corrupted data in a (computer) system. For example, a fault in a DRAM cell will never be observed if the memory location is never accessed. Specifically, an error is a deviation from accuracy or correctness.
failure	the incorrect functioning of a system as perceivable by a user or the system's environment as a consequence of an error. A failure is the non-performance, the untimely performance or the performance in a subnormal quantity or quality of some action that is due or expected.

redundant hardware	additional hardware for performing the same function as another hardware part, provided that in faultless operation you could renounce on either hardware parts of the system without losing functionality.
data representation	a physical or logical encoding (scheme) for data, which allows the latter to be processed, stored or transmitted by a machine.
redundancy in data representation	a representation of data using more resources than strictly necessary to encode the desired information such that in the error free situation one could renounce some of said resources without losing information.
redundancy in operation	performing (a set of) operations more than once, performing sequentially different implementations of a particular function or performing additional operations which (allow to) restore a system in a state from which its correct operation can be resumed after a failure.
time diversity	the concept to have a redundant system in which one of the redundant components operates with a delay with respect to the other in order to avoid common mode failures that would affect both redundant components in the same way at the same time, thereby not being detectable/correctable.

master-checker setup	a redundant configuration in which a master CPU drives the system. The checker CPU is synchronized (often at clock level) with the master. It processes the input data stream as the master (and often also the very same program). Whenever the master drives an output signal, the checker compares its own value with the data written by the master. A mismatch triggers an error signal. The master-checker mode is supported in many modern microprocessors by a comparator integrated into the pin driver circuitry, thus reducing the external logic to a few chips for interfacing the error signals. The master-checker system generally gives more accurate answers by ensuring that the answer is correct before passing it on to the application requesting the algorithm being completed. It also allows for error handling if the results are inconsistent. Depending on the merit of a correct answer, a checker-CPU may or may not be warranted. In order to alleviate some of the cost in these situations, the checker-CPU may be used to calculate something else in the same algorithm, increasing the speed and processing output of the CPU system. There are two possible configurations: Master-Listener and Cross-Coupled. The Master-Listener lock step configuration pairs two processors, with one as a complete Master and the other as a complete Listener, the latter having disabled output drivers. In the Cross-Coupled configuration, one of the processors, the SI-Master, drives the system interface bus, while the other processor, the SC-Master, drives the secondary cache bus. The SI-Master has disabled output drivers for the secondary cache interface bus while the SC-Master has disabled output drivers for the system interface bus.
normal operating mode	the operation of a system or software once it is deployed and provides the desired service as opposed to its development, maintenance, test or idle time.
fault masking	hiding the presence of a fault to the user or the environment of a (computer system by means of some sort of redundancy such that the perceived system functionality is not affected.
active fault masking	taking particular actions (e.g. reconfiguration or failover) not performed in the error free situation to mask a fault.
passive fault masking	when a system operates such that no particular action is necessary to mask a fault because all necessary operations are constantly performed independently of the presence of a fault (e.g. majority voting).
interconnection	physical media and may be of point-to-point type or of bus type. Two interconnections are only considered redundant if: they both physically connect the same nodes, wherein nodes are the source producing or the final destination consuming the data to be transmitted, and are configured to perform the same data transmissions.

monitoring	observing and/or measuring parameters or status of a running system.
mirrored data	two copies of the data where it is supposed that both copies contain the same data at any moment.
backed up data	the second copy of the data reflects the data of the first copy at a particular moment.

G06F 11/36

Definition statement

Replace: The existing Definition statement with the following updated statement.

Evaluating a program's functionality to find and fix bugs or abnormalities with the intent to check the program against its specifications and identify any inconsistencies using a variety of techniques, such as unit testing, integration testing and system testing.

Tools frequently used to identify the source of problems and ensure the software functions as intended are as follows (non-exhaustive):

- debuggers;
- profilers; and
- log analysers.

References

Informative references

Insert: The following new row into the existing Informative references table.

Arrangements for software engineering	G06F8/00

Replace: The text in the following row of the existing Informative references table with the following updated text.

Detection or location of defective computer hardware by testing during standby operation or during idle time	G06F11/22

Glossary of terms

Replace: The text in the following row in the existing Glossary of terms table with the following updated text.

bug	a bug in a program results from a faulty implementation and produces an incorrect or unexpected result or causes the program to behave in unintended ways.

G06F 11/3604

Definition statement

Replace: The existing Definition statement with the following updated statement.

Analysis of software, i.e. verifying that a program or its specification satisfies certain properties without involving testing, i.e. no test inputs are provided to the program under analysis. For example, it may involve scanning the source code and analysing the dependencies between the various components, or the use of certain variables. It may involve the use of formal methods, like model checking or theorem proving which aim to formally guarantee certain properties, for example that the program is well-typed or deadlock free. In principle, analysis of software does not require the program to be executed (static analysis), but in some cases the program is executed, for example, to verify runtime properties (dynamic analysis).

References

Insert: The following new Informative references section.

Informative references

Attention is drawn to the following places, which may be of interest for search:

Program documentation	G06F8/73
Structural analysis for program understanding	G06F8/75
Software metrics	G06F8/77

G06F 11/362

Definition statement

Replace: The existing Definition statement with the following updated statement.

The activity of identifying, evaluating and fixing bugs that are found during runtime execution or testing.

Delete: The entire Informative references section.

G06F 11/3668

Definition statement

Replace: The existing Definition statement with the following updated statement.

The process of executing a program or a discrete program unit, using test inputs with the intent of finding errors.

References

Informative references

Replace: The text of the following row in the existing Informative references table.

Detection or location of defective computer hardware by testing during standby operation or during idle time	G06F11/22

G06F 11/3698

Definition statement

Replace: The existing Definition statement with the following updated statement.

Environments, frameworks, graphical user interfaces [GUI] or simulators that aim at supporting or facilitating the task of a user during the various phases of analysing, testing or debugging of software, e.g. to navigate into the code, to add or remove breakpoints, to visualise execution traces or to edit/maintain/archive test suites.

The environment may be comparable to a software development environment [SDE], but it contains features that are specific to the phases of analysing, testing or debugging of software.