Serial No. 363,091

Filing Date 14 December 1994

Inventor John J. McGarry

NOTICE

The above identified patent application is available for licensing. Requests for information should be addressed to:

OFFICE OF NAVAL RESEARCH
DEPARTMENT OF THE NAVY
CODE OOCC3
ARLINGTON VA 22217-5660

Navy Case No. 74966

14 Dec 94

QUANTITATIVE SOFTWARE DEVELOPMENT ASSESSMENT

STATEMENT OF GOVERNMENT INTEREST

The invention described herein may be manufactured and used by or for the Government of the United States of America for Governmental purposes without the payment of any royalties thereon or therefor.

BACKGROUND OF THE INVENTION

(1) Field of the Invention

The present invention relates generally to the field of process management. In particular, it is a method for quantitatively measuring software processes and products using metrics describing a process.

(2) Description of the Prior Art

In many commercial settings, the evolution of quantitative assessment methodologies has led to increased productivity, better resource management, and higher quality products.

Some attempts have been made in the prior art to apply these quantitative methodologies in the field of software development. However, the inherent characteristics of software makes the application of these principles difficult. Specifically, quantitative measurement of software development processes and products is made difficult by the volatility of these projects,

the significant effects of interrelated requirements and constraints, and the difficulty of accurately quantifying measures of both the amount of software completed and the quality of the completed software. These difficulties often produce inconsistent and sometimes erroneous results. These weaknesses have prevented the application of quantitative assessment techniques in many commercial development programs. As such, there is little objective management and technical data available to support development process control and quality assessment.

SUMMARY OF THE INVENTION

Accordingly, it is an object of the present invention to effectively evaluate the software development process and related software products and to generate objective management and technical data.

It is a further object of the present invention to provide an overall assessment that quantifies and integrates objective measures of software development attributes into an aggregate project profile.

It is a further object of the present invention to be sensitive to the cost of the quantitative measurements required during the use of the method.

A still further object of the present invention is to produce assessment results that can be readily validated, that are applicable across multiple software development projects, and that are consistent for all projects.

Yet another object of the present invention is to complement the volatile nature of software development by integrating diverse attribute data and relating both software process and product issues under a cause-effect relationship framework.

In accordance with these and other objects, a method for monitoring, measuring, and controlling the evolution of a software development project is provided. The method includes software assessment processes, tools, and techniques focused on the evaluation of the software development processes, development progress, development resource application, and software product quality. In particular, the method is based on a software development assessment structure which includes defined measures of software process and product attributes within the context of the software development program constraints, characteristics, and limitations. The structure integrates software attribute measures in the general categories of resource application, development process, and product quality. It incorporates measurement and evaluation approaches which can be applied during all phases of the software development life cycle, and which can be tailored to specific program characteristics and overall program management and technical objectives.

The general method of the invention includes defining software issues, measuring software attributes and generating indicators thereto, and performing a quantitative assessment of these indicators. The specific method of the invention includes identifying and prioritizing software issues, mapping those

issues to effective measures, defining the measurement requirements for software attributes, developing methodologies for performing measurements of the attributes, performing, managing and collecting those measurements, defining and correlating software indicators, clarifying and evaluating issues with respect to the indicators, correlating process factors with product factors, and generating recommendations based on the correlated factors.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing objects and other advantages of the present invention will become fully understood from the following detailed description and reference to the appended drawings, wherein:

FIG. 1 is a top level process flow chart for the quantitative software assessment method;

FIG. 2 is a depiction of the key interrelations between software development schedule, resources, capability and development performance;

FIG. 3 is a listing of commonly measured software attributes for a typical software development project;

FIG. 4 is a depiction of a software indicator; and

FIG. 5 is a flow chart for the process-product analysis sequence.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring now to the drawings, and in particular to FIG. 1, a top level process flow chart for the quantitative software assessment method 10 is provided. The method incorporates a sequential process consisting of four phases. The first phase, Software Issue Definition 11, encompasses the identification and prioritization of software development issues and the creation of mappings between these issues and effective measures to generate attributes quantifying each issue. The second phase, Software Attribute Measurement 13, encompasses the definition, measurement, and tracking of software process and product attributes defined in the first phase. The third phase, Software Indicator Generation 15, encompasses the instantiation of quantitative analysis products and related software measurement attributes. The final phase, Software Quantitative Assessment 17, encompasses the integrative evaluation of the assembled attributes using multiple tools and techniques within the context of the developmental program objectives, constraints and characteristics. Throughout the process, software process and product attributes are interrelated within the assessment structure to identify process related software quality impacts and to identify corrective actions necessary for improvement. All four phases of the assessment method are specifically structured to meet several criteria and share several properties.

First, the method is based upon defined quantitative measures of software process and product attributes measured

consistently by a defined measurement methodology during the life cycle of the product. The method is also flexible and tailorable to distinct software development program characteristics, objectives and limitations. In order to achieve this level of customization, the method encompasses a variety of commercial measurement and assessment tools including data generation utilities, software development process models, metrics databases and utilities, attribute assessment matrices, software product analyzers, and graphics display interfaces.

The method is assessment driven in that specific issues and concerns drive the applied software attribute measures and the analysis focus. The measures selected are specifically chosen to be non-constrictive. Within the overall assessment method, different measures can be applied for different projects. However, each class of measurement is defined and applied consistently across the development. The consistency of the method with respect to a given project, the use of multiple classes of measurement and the use of substantive qualitative engineering observations ensure valid and objective analysis results.

Finally, multiple possible target values for each attribute are tracked corresponding to separate baseline possibilities. Initially, software attribute measurement results are analyzed on an attribute by attribute basis, with the actual measured values for each attribute compared to the possible target values. These individual attributes are next integrated into an overall profile

of the development process and products. These integrative profiles are used for project tracking and valid cross-project comparisons.

Referring now to FIG. 2, a depiction of the key inter-relations between software development schedule, resources, capability and development performance 21 is provided. These elements are key issues within the context of the Software Issue Definition phase 11 of the Quantitative Software Assessment method 10. The Software Issue Definition phase 11 is the initial analysis process in the Quantitative Software Assessment method. It is first implemented during the planning phase prior to program implementation and continues as the development process proceeds and software products are designed, developed, tested, and released. The objective of the Software Issue Definition phase 11 is to identify and prioritize the software process and product issues so that measurement and analysis efforts can be focused and cost effective. It encompasses issue identification, issue prioritization, and issue to measures requirements mapping. Issues are initially defined based upon the schedule 22, resource 24, and technical (software reuse 26 and software process 28) characteristics of a particular software development program, and the constraints defined in the relationships between these characteristics. Factors 29, such as those shown in FIG. 2, can impact both the software function capability as well as the development productivity, cost per unit of the product, and the final product quality. Although the

Software Issue Definition phase is shown in FIG. 1 as the first phase of the complete system, it is periodically repeated throughout the product life cycle to update existing development issues and to identify and prioritize new issues.

Referring now to FIG. 3, a summary of commonly measured software process attributes 31 and software product attributes 32 are provided. These attributes are identified during the Software Issue Definition phase through a variety of commercial applications including the Software Life Cycle Model (SLIM) and SLIM control packages from Quantitative Software Management, Inc., the System Evaluation and Estimation of Resources and Software Estimation Model packages from Galorath Associates, the Goal-Question-Metric Paradigm from the University of Maryland, the Objectives-Principles-Attributes Paradigm from the Virginia Polytechnic Institute, SASET from the U.S. Navy, the Software Capacity Maturity Model from the Software Engineering Institute, and the public domain Software Constructive Cost Model.

Once the specific attributes are identified, they are recorded individually during the Software Attribute Measurement phase B. Each attribute is recorded using a defined measurement methodology. Although the set of attributes required are flexible and tailored to each specific development program, the methodologies for taking the required attributes are strict and well-defined. The strict methodologies insure quantitative consistency within the context of the software development program, and even across programs. Measurements are taken from

numerous sources throughout the developmental life cycle and can be either manually or automatically measured. The attributes are stored in a metrics database and can be accessed and manipulated through a variety of commercial tools including Oracle database utilities, software product attribute measurement tools, developer financial management methods, project schedule planning and tracking methods (PERT), and a variety of CASE tools and Defect database utilities.

Referring now to FIG. 4, a graphical depiction of a software indicator is provided as an example. A single software indicator, size expressed in lines of code (the ordinate) over time, is depicted. Total planned code 41 is shown along with new planned code 42 and new actual code 43. An indicator may be generated from an attribute, for example, code size or growth, defect level, etc. Indicators such as these are the output from the third phase of the method, the Software Indicator Generation phase 15. In particular, the Software Indicator Generation phase 15 renders information from the data collected during the Software Attribute Measurement phase 13 into a form that allows project managers to easily ascertain progress towards goals, clarify those goals, and to plan for new contingencies. The software indicators generated during this phase are based on both individual attributes and aggregate measures. These measures are graphically rendered using the previously collected data (stored in a metrics database), graphics capable workstations, and commercial or public domain graphics generation or reporting packages.

The Software Quantitative Assessment phase 17 is a key structure in the complete software assessment method. In particular, during the Software Quantitative Assessment phase 17, existing software development issues are quantitatively clarified, new or possible software development issues are identified, the degree of impact of a given software development issue is evaluated, process and product attributes are correlated, and recommendations for improvement are generated. To achieve these goals, the data provided by all of the other phases of the method are integrated into an overall profile of the software development program during this phase. This overall profile encompasses the quantitative findings within a context of software engineering principles and specific program characteristics and observations. Once the overall profile has been generated, components that demonstrate the highest degree of development cost, schedule or technical risk are identified and isolated by comparing attributes generated from multiple attribute level indicators and by determining which attributes affect a number of aggregate measures over a period of time. The overall profiles are also evaluated in the context of the cause and effect relationships between the software development process and resultant software products. Development constraints which can significantly impact the integrity, efficiency, and quality of the software product are identified.

Referring now to FIG. 5, a flow chart for the process-product analysis sequence is depicted. The Software Quantitative

Assessment phase 17 encompasses this sequence. First, as progress is made, individual product and process attributes 51 are measured to establish the size, effort levels, and cost of the project. Later in the product life cycle, attributes 53 provide information on productivity and the number of defects in the product. This information allows resources to be effectively allocated to products and a level of stable productivity 55 to be achieved. Finally, a resultant product 56 is generated with a high level of quality and consistency.

The advantages and new features of the present invention are numerous. The invention provides a consistent evaluation approach which can be tailored to many different types of software development. It is flexible enough to insure cost-effective implementation. The structure incorporates quantitative analysis which clearly identifies the causes of process and product deficiencies. The structure also provides, based upon software attribute data characteristics, the ability to project key software development process issues and related product quality impacts prior to product generation. Most significantly, the process supports the identification of key software development issues and risk areas based upon the integration and evaluation of diverse software attributes.

It will be understood that many additional changes in the details, materials, steps and arrangement of parts, which have been herein described and illustrated in order to explain the nature of the invention, may be made by those skilled in the art

within the principle and scope of the invention

Navy Case No. 74966

QUANTITATIVE SOFTWARE DEVELOPMENT ASSESSMENT

ABSTRACT OF THE DISCLOSURE

A software method for monitoring, measuring, and controlling the evolution of a software development project is provided. The method compares quantitative measures of software product and process attributes with expected and observed product characteristics over the development life cycle. The resulting attribute measurements are evaluated in the context of an overriding issue definition that identifies and prioritizes software product and process issues. The method includes a set of software products which can be utilized to implement the method.

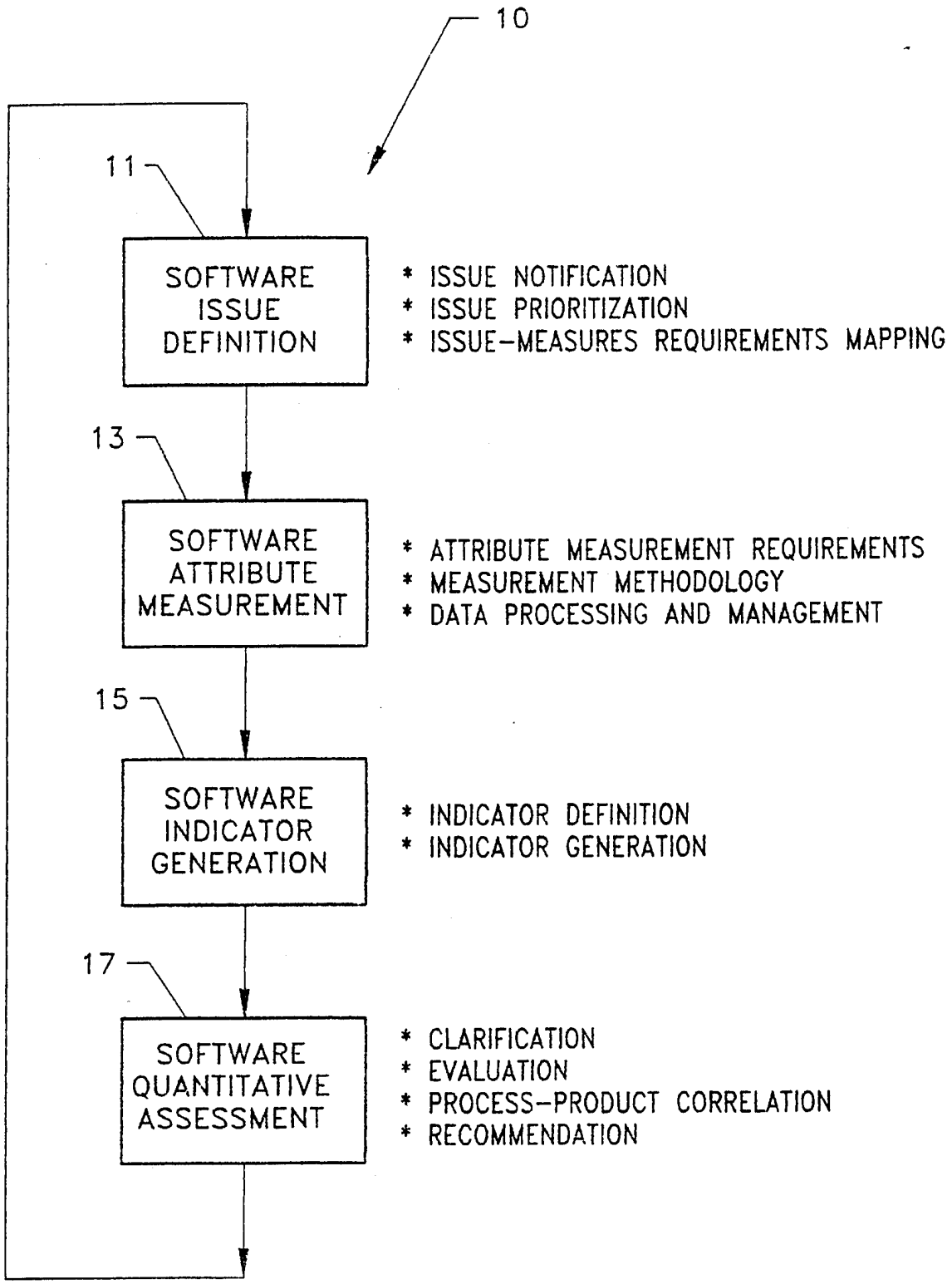
10
11
SOFTWARE ISSUE DEFINITION
* ISSUE NOTIFICATION
* ISSUE PRIORITIZATION
* ISSUE-MEASURES REQUIREMENTS MAPPING
13
SOFTWARE ATTRIBUTE MEASUREMENT
* ATTRIBUTE MEASUREMENT REQUIREMENTS
* MEASUREMENT METHODOLOGY
* DATA PROCESSING AND MANAGEMENT
15
SOFTWARE INDICATOR GENERATION
* INDICATOR DEFINITION
* INDICATOR GENERATION
17
SOFTWARE QUANTITATIVE ASSESSMENT
* CLARIFICATION
* EVALUATION
* PROCESS-PRODUCT CORRELATION
* RECOMMENDATION

FIG. 1

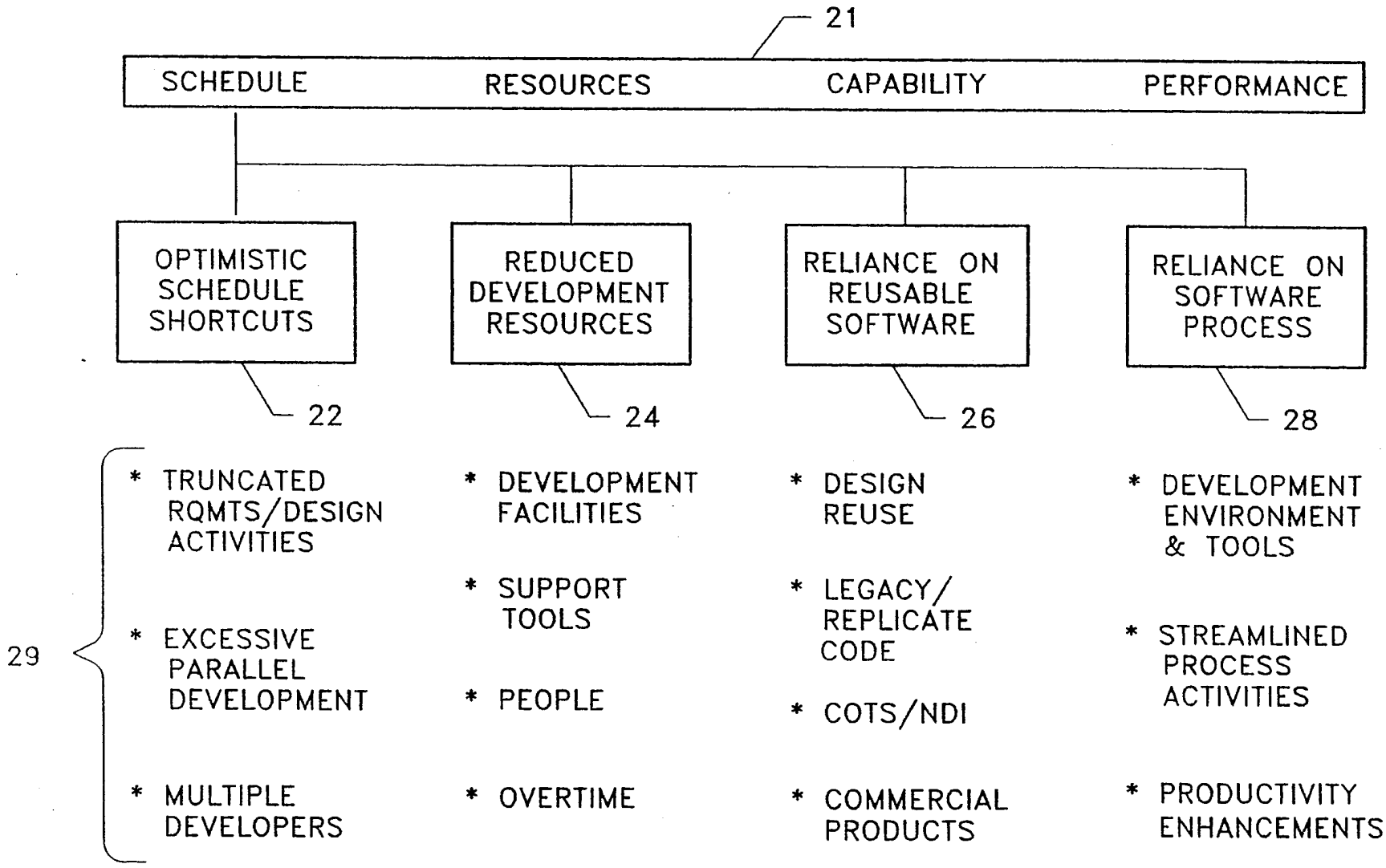
21
SCHEDULE
RESOURCES
CAPABILITY
PERFORMANCE
OPTIMISTIC SCHEDULE SHORTCUTS
REDUCED DEVELOPMENT RESOURCES
RELIANCE ON REUSABLE SOFTWARE
RELIANCE ON SOFTWARE PROCESS
22
24
26
28
29
* TRUNCATED RQMTS/DESIGN ACTIVITIES
* EXCESSIVE PARALLEL DEVELOPMENT
* MULTIPLE DEVELOPERS
* DEVELOPMENT FACILITIES
* SUPPORT TOOLS
* PEOPLE
* OVERTIME
* DESIGN REUSE
* LEGACY/ REPLICATE CODE
* COTS/NDI
* COMMERCIAL PRODUCTS
* DEVELOPMENT ENVIRONMENT & TOOLS
* STREAMLINED PROCESS ACTIVITIES
* PRODUCTIVITY ENHANCEMENTS

FIG. 2

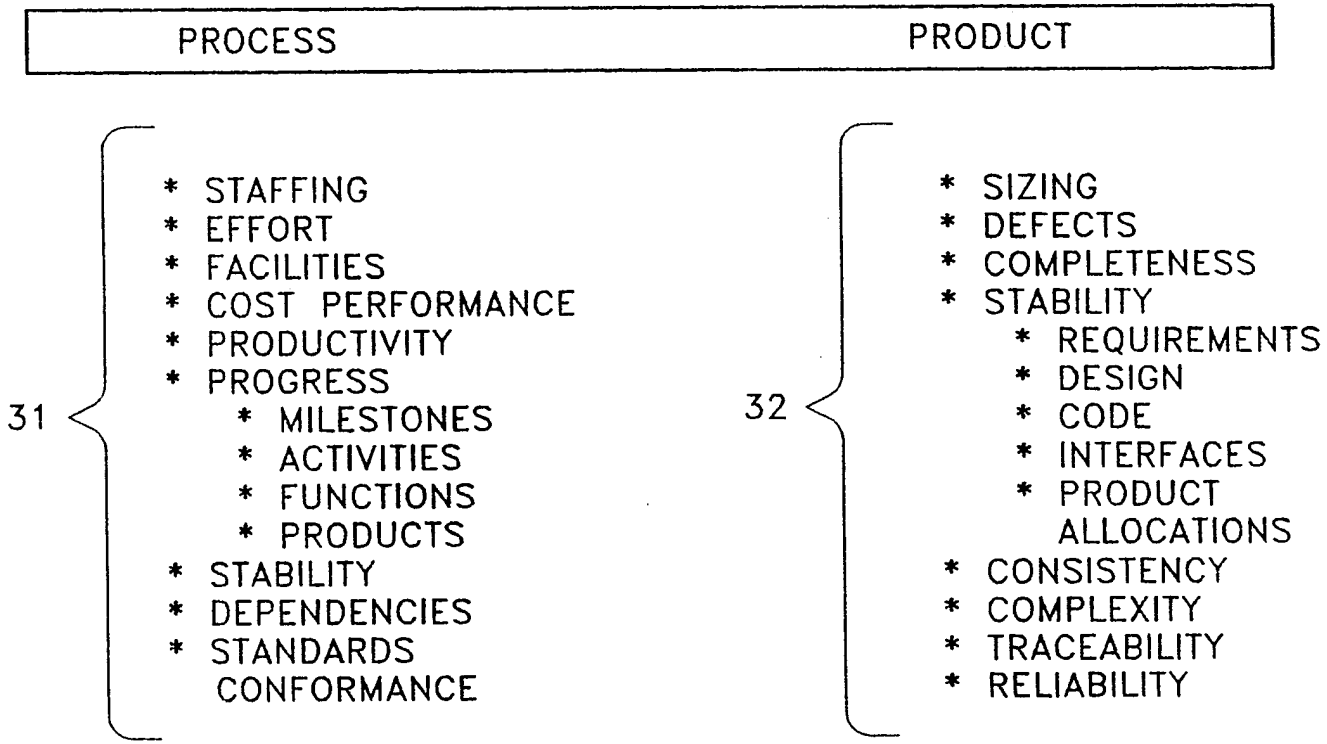
PROCESS
PRODUCT
31
* STAFFING
* EFFORT
* FACILITIES
* COST PERFORMANCE
* PRODUCTIVITY
* PROGRESS
* MILESTONES
* ACTIVITIES
* FUNCTIONS
* PRODUCTS
* STABILITY
* DEPENDENCIES
* STANDARDS CONFORMANCE
32
* SIZING
* DEFECTS
* COMPLETENESS
* STABILITY
* REQUIREMENTS
* DESIGN
* CODE
* INTERFACES
* PRODUCT ALLOCATIONS
* CONSISTENCY
* COMPLEXITY
* TRACEABILITY
* RELIABILITY

FIG. 3

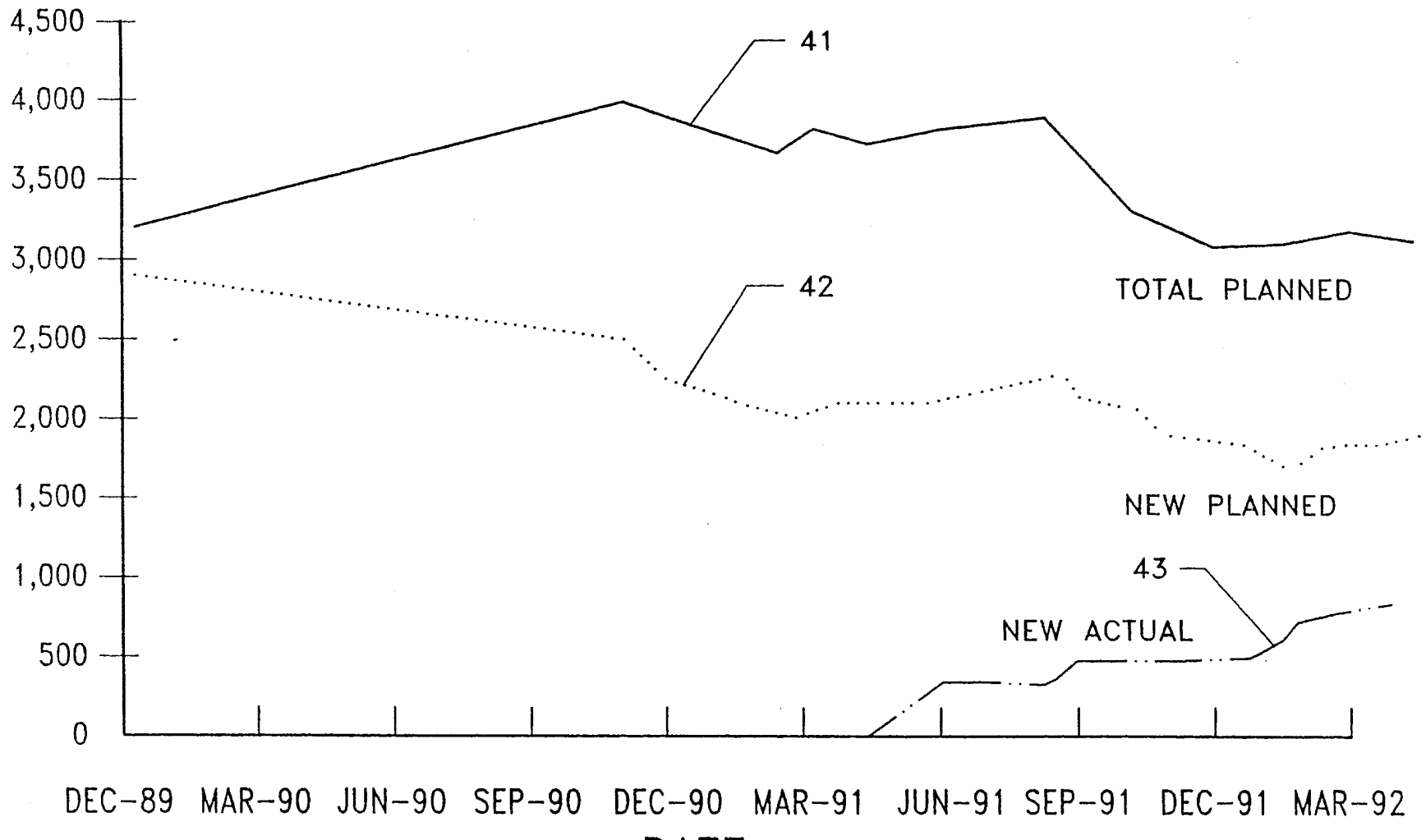
4,500
4,000
3,500
3,000
2,500
2,000
1,500
1,000
500
0
41
42
43
TOTAL PLANNED
NEW PLANNED
NEW ACTUAL
DEC-89
MAR-90
JUN-90
SEP-90
DEC-90
MAR-91
JUN-91
SEP-91
DEC-91
MAR-92
DATE

FIG. 4

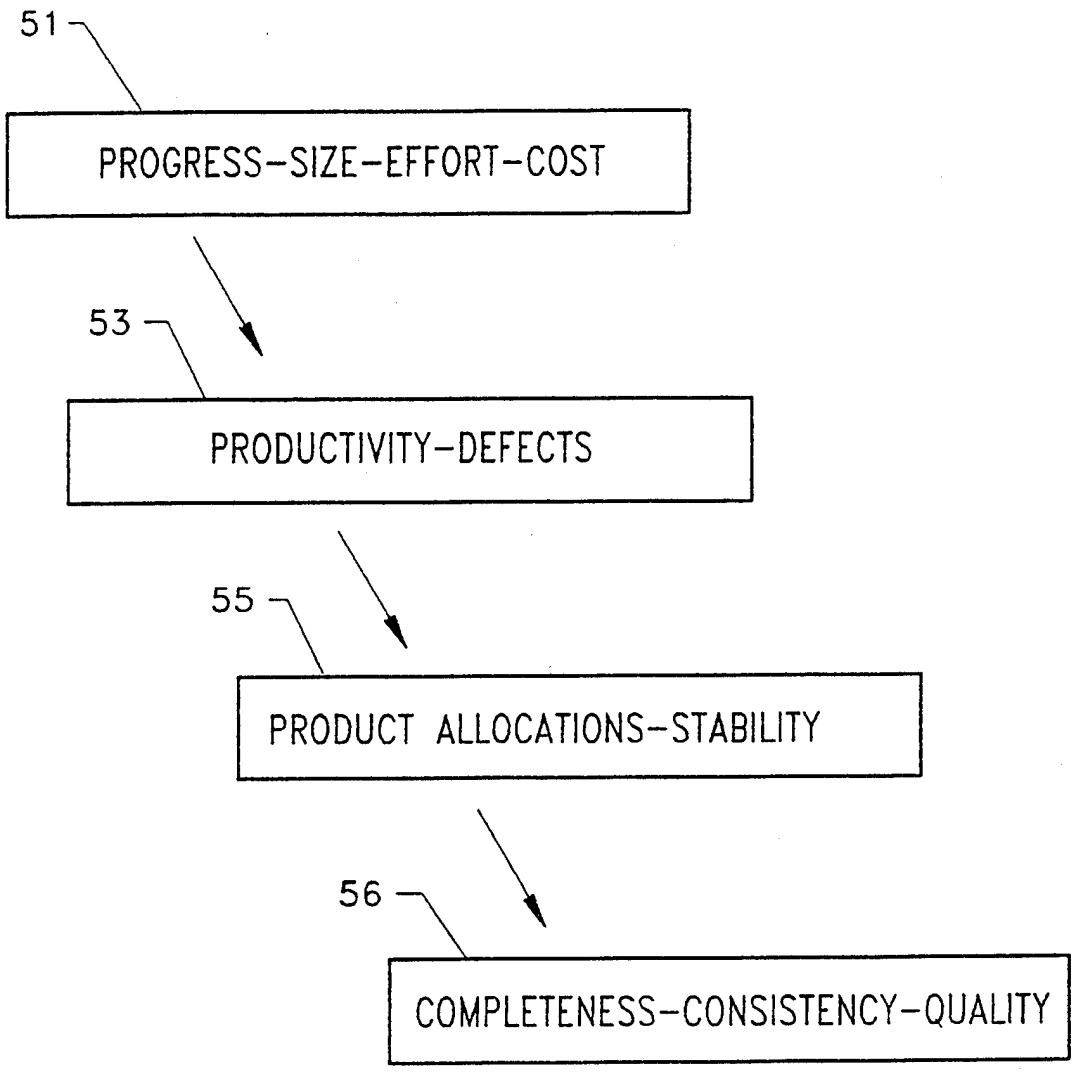
51
PROGRESS-SIZE-EFFORT-COST
53
PRODUCTIVITY-DEFECTS
55
PRODUCT ALLOCATIONS-STABILITY
56
COMPLETENESS-CONSISTENCY-QUALITY

FIG. 5