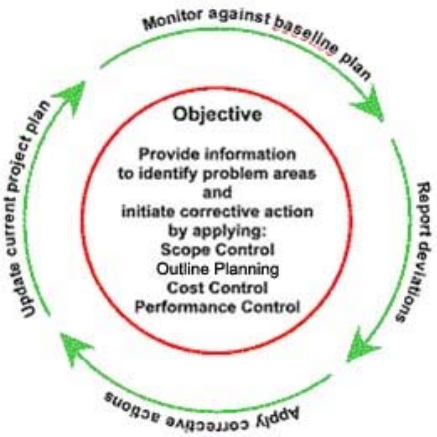
Answer Question No. 1 ;

This report includes many organizational techniques that have proved to be very helpful. By combining new and traditional planning techniques the overall project should run exceptionally smooth. Basically, this report was developed to create an excellence organizational payroll system format.

The payroll system in a certain company may be described as follows:-



Payroll Diagram

Managing, by the way was well defined as long ago as 1916 by Henri Fayol. He said:

"To manage is to forecast and plan, to organize, to command, to coordinate and to control. To forecast and plan means examining the future and drawing up the plan of action. To organize means to build up the dual structure, material and human, of the undertaking. To command means maintaining activity amongst the personnel. To

coordinate means bonding together, unifying and harmonizing all activity and effort. To control means seeing that everything occurs in conformity with established rule and expressed command." (Henri Fayol, Administration Industrielle et Generale, 1916.)

And:

Project Management is ;

"The art of directing and coordinating human and material resources throughout the life of a project by using modern management techniques to achieve predetermined objectives of scope, cost, time, quality and participant satisfaction."

Not everyone is familiar with the word "Scope" by the way. "Scope" for payroll means the work content and finished "products" for which the project has been designed. Sometimes scope may be represented by a statement of the results or performance expected, leaving the content details to the designer. Similarly each phase content, or component such as "work package" also have associated scopes.

Scope is fully defined by detailing the end products resulting from the project, including quality standards, all activities performed and the resources consumed. A scope

statement should be introduced by a brief background to the project, or component, and the general objective.

Project Manager's Objectives ;

It follows from the foregoing that the Project Manager's personal objectives must be to:

- Attain the willing commitment of people to assigned tasks;
- Achieve the coordination and collaboration of different work groups, responsibility centers, and entire organizations, including those of the owner;
- Achieve cooperation by placing a high premium on reliability and timeliness of information, and by discouraging unnecessary or irrelevant information;
- Steer the project to completion in an orderly and progressive manner;
- Ensure that trade-offs between scope, cost and time are satisfactory and acceptable, and are seen to be so; and
- Perpetuate development of personal and professional skills and the potentialities of project participants.

Process of management control ;

Plan, Organize, Execute, Monitor and Control :

The basic process of management control can easily be remembered by the mnemonic POEM standing for Plan, Organize, Execute, Monitor and Control.

Plan **-** The first step is to plan the project with respect to scope, time and cost. What precisely is to be done? Why? If it is, say, a new plant, what is the purpose and process in the plant? How is the job to be done? Why should the project be done one way rather than another? Indeed, why should it be done at all? Where is it to be built? Who will design and construct it? What resources in terms of materials, manpower, finances and time are required? What risks are involved? What strategies are required to deal with unplanned occurrences?

Organize **-** The second basic step is an extension of the planning process. A careful analysis must be made of the various activities required in planning and executing a project, to provide a closely related project team structure. For every project activity

(e.g. programming, estimating, design, planning, procurement, construction) there must be a very clear definition of who is responsible, and who has the authority to execute the activity. The system must have a very clear definition of the scope, cost and time budget for that activity.

Execute **-** The methods by which the Plan is executed or implemented are critical. No project manager (or other member of the project team) will be successful unless he understands the basic needs of human beings, their strengths and weaknesses, mental and social abilities, and how to weld a complex mixture of humans into a dynamic and productive team. The single most important characteristic of a successful project manager is his ability to manage people.

Monitor and Control **-** Continued monitoring, reporting and forecasting must take place during project implementation, and the forecasts compared to the Plan. Deviations must immediately receive management attention, either by reallocation of resources or modifications to the Plan (with the client's approval if his objectives are affected). Without a detailed Plan, there is no basis for comparison, no determination of deviation, and hence no satisfactory basis for corrective action. Clearly then, a successful project management system is one which monitors and responds by a control action as early as possible after an event.

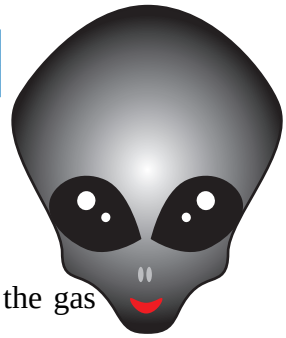
The diagram illustrates the elements of management control by outlining the general activities which may be expected within each stage of the project management process in various project situations.

Elements of Control ;

As noted earlier, project management relies heavily on the science of systems. A practical example will help in the understanding of a control system. In a simple machine-to-machine system such as an air conditioner, the input is the electric power and the output is cold air. For this we need three essential control tools:

- A monitoring mechanism, in this case, a thermostat;
- A comparative device, e.g., the thermostat signal with a set point or objective; and
- A preset formula and a means for sending a corrective signal.

The preset formula and corrective signal in its simplest form is on-off. Obviously, other more sophisticated formulae and signals are possible. This can be seen in a man-to-



machine system such as an automobile where graduated control is exercised by the gas and brake pedals.

Project Management is a man-to-man system. In this case the input is essentially design information and resources of materials and labor. The output is a completed facility. The processing is done by designers, draftsmen, skilled labor, etc., who transform the raw data through drawings to contracts to construction and finally to project start-up.

Control is exercised through monitoring,implementing,review,reporting and forecasting the output, comparing this to the project objectives and sending corrective signals to the input of data and resources. Thus the output is made to conform closely to the objectives operation. This cycle is illustrated in *Figure 2* .

The Post Audit Review Approach;

Consider the key risks for the audit to be those concerned with payroll recognition and systems, and returns. Conducted audit work on the returns received from the branches by head office to confirm the reliability of the information contained in them.

The work focussed on ;

• Documenting systems and controls over payroll streams

• Substantive testing of income

• Agreeing returns received from the branches to underlying books and records and reviewing the systems and controls in operation implementation.

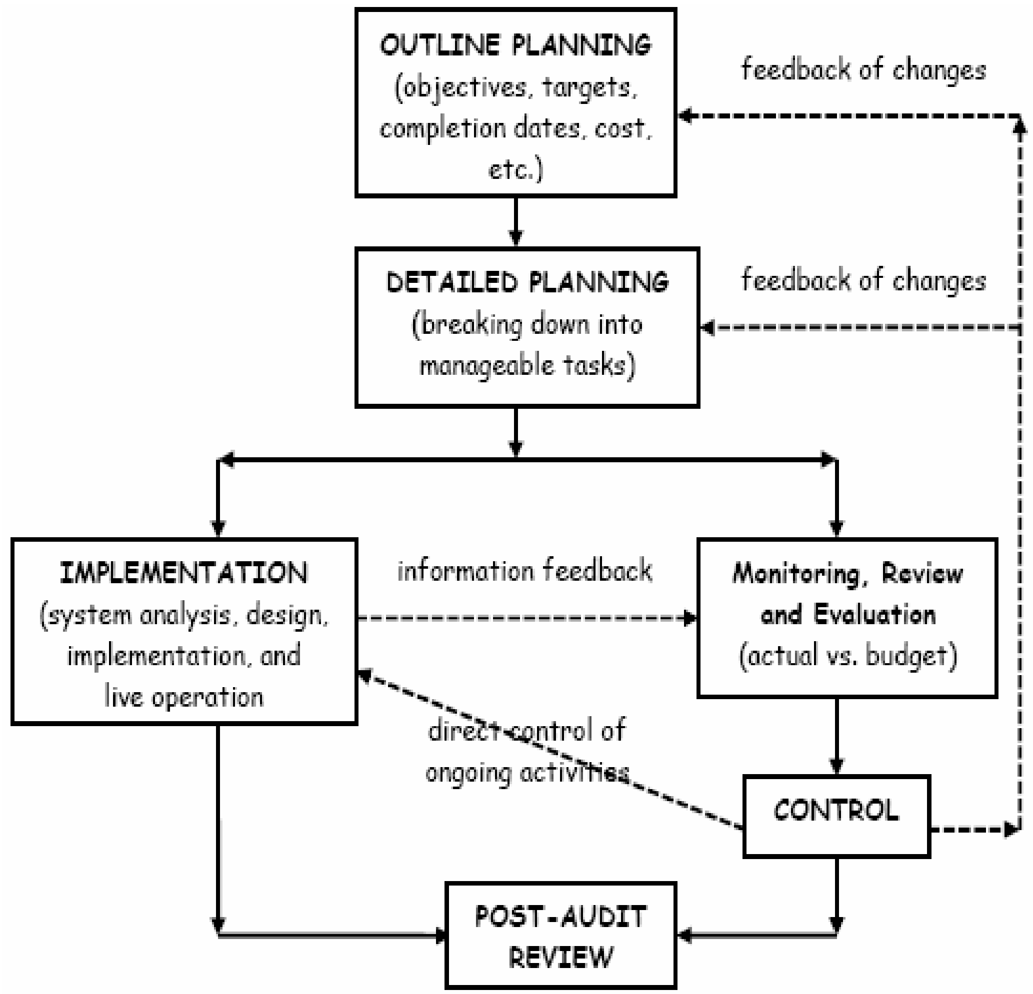


Figure 2

Answer Question 2;

Object-oriented design and development has become popular in today's software development environment. The benefits of object-oriented software development are now widely recognized. Object-oriented development requires not only different approaches to design and implementation; it also requires different approaches to software metrics.

The metrics for object-oriented systems are different due to the different approach in program paradigm and in object-oriented language itself. An object-oriented program paradigm uses localization, encapsulation, information hiding, inheritance, object abstraction and polymorphism, and has different program structure than in procedural languages.

Software metrics are often categorized into *product metrics* and *design metrics*. Project metrics are used to predict project needs, such as staffing levels and total effort. They measure the dynamic changes that have taken place in the state of the project, such as how much has been done and how much is left to do. Project metrics are more global and less specific than the design metrics.

Design metrics are measurements of the static state of the project design at a particular point in time. These metrics are more localized and prescriptive in nature. They look at the quality of the way the system is being built.

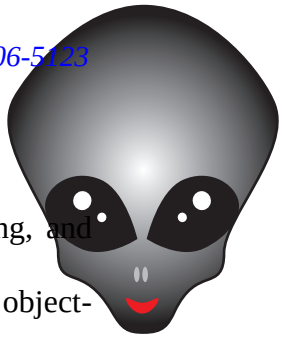
Design metrics can be divided into *static metrics* and *dynamic metrics*. Dynamic metrics have a time dimension and the values tend to change over time. Thus dynamic metrics can only be calculated on the software as it is executing. Static metrics remain invariant and are usually calculated from the source code, design, or specification.

There are quite a few sets of proposed metrics of object-oriented metrics for object-oriented software in the literature and research papers.

Rationale for measurement ;

The intent of the metrics proposed is to provide help for object-oriented developers and managers to foster better designs, more reusable code, and better estimates. The metrics should be used to identify anomalies as well as to measure progress. The numbers are not meant to drive the design of the project's classes or methods, but rather to help us focus our efforts on potential areas of improvement.

The metrics are guidelines and not rules and they should be used to support the desired motivations. The intent is to encourage more reuse through better use of abstractions and division of responsibilities, better designs through detection and



correction of anomalies. Positive incentives, improvement training and mentoring, and effective design reviews support probability of achieving better results of using object-oriented metrics.

For example, C++ tends to have larger method sizes than Smalltalk. Thresholds are not absolute laws of nature. They are heuristics and should be treated as such. Possible problems in the system designs can be detected during the development process.

Software should be designed for maintenance. The design evaluation step is an integral part of achieving a high quality design. The metrics should help in improving the total quality of the end product, which means that quality problems could be resolved as early as possible in the development process. It is a well-known fact that the earlier the problems can be resolved the less it costs to the project in terms of time-to-market, quality and maintenance.

Code and design metrics suite ;

Metric 1: Weighted Methods per Class (WMC)

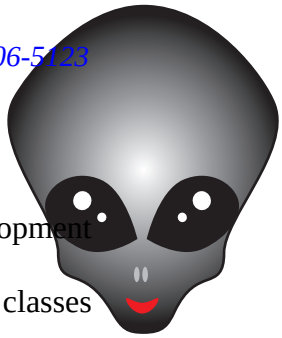
WMC is a sum of complexities of methods of a class. Consider a Class C_1 with Methods $M_1 \ldots M_n$ that are defined in the class. Let $c_{1 \ldots} c_n$ be the complexity of the methods[1]. Then:

$$\mathrm{WMC} = \sum_{i=1}^{n} c_i$$

WMC measures size as well as the logical structure of the software. The number of methods and the complexity of the involved methods are predictors of how much time and effort is required to develop and maintain the class. The larger the number of methods in a class, the greater the potential impact on inheriting classes. Consequently, more effort and time are needed for maintenance and testing. Furthermore, classes with large number of complex methods are likely to be more application specific, limiting the possibility of reuse. Thus WMC can also be used to estimate the usability and reusability of the class. If all method complexities are considered to be unity, then WMC equals to *Number of Methods* (NMC) metric.

Evaluation of metrics ;

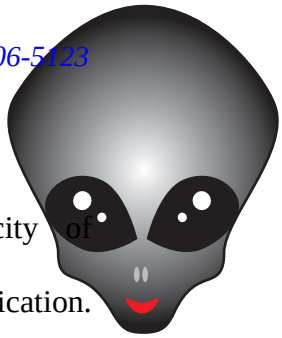
Chidamber and Kemerer who introduced the basic suite for collecting object-oriented code and design metrics tested the metrics suite with two projects.



Site A is a software vendor that uses object-oriented design in their development work and has a collection of different C++ class libraries. Metrics data from 634 classes from two C++ class libraries that are used in the design of graphical user interfaces (GUI) were collected. Both these libraries were used in different product applications for rapid prototyping and development of windows, icons and mouse based interfaces. Reuse across different applications was one of the primary design objectives of these libraries. These typically were used at Site A in conjunction with other C++ libraries and traditional C-language programs in the development of software sold to UNIX workstation users.

Site B is a semiconductor manufacturer and uses the Smalltalk programming language for developing flexible machine control and manufacturing systems. Metrics were collected on the class libraries used in the implementation of a computer aided manufacturing system for the production of VLSI (Very Large Scale Integration) circuits. Over 30 engineers worked on this application, after extensive training and experience with object orientation and the Smalltalk environment. Metrics data from 1459 classes from Site B were collected.

The data from two different commercial projects and subsequent discussions with the designers at those sites lead to several interesting observations that may be useful for managers of object-oriented projects. Designers may tend to keep the inheritance



hierarchies shallow, forsaking reusability through inheritance for simplicity or understanding. This potentially reduces the extent of method reuse within an application. However, even in shallow class hierarchies it is possible to extract reuse benefits, as evidenced by the class with 87 methods at Site A that had a total of 43 descendants. This suggests that managers need to proactively manage reuse opportunities and that this metrics suite can aid this process.

Another demonstrable use of these metrics is in uncovering possible design flaws or violations of design philosophy. As the example of the command class with 42 children at Site A demonstrates, the metrics help to point out instances where sub classing has been misused. This is borne out by the experience of the designers interviewed at one of the data sites where excessive declaration of sub classes was common among engineers new to the object-oriented paradigm. These metrics can be used to allocate testing resources. As the example of the interface classes at Site B (with high CBO and RFC values) demonstrates, concentrating test efforts on these classes may have been a more efficient utilization of resources.

Conclusions ;

The need for such metrics is particularly acute when an organization is adopting a new technology for which established practices have yet to be developed. The metric suite is not adoptable as such and according to some other researches it is still premature to begin applying such metrics while there remains uncertainty about the precise definitions of many of the quantities to be observed and their impact upon subsequent indirect metrics. For example the usefulness of the proposed metrics, and others, would be greatly enhanced if clearer guidance concerning their application to specific languages were to be provided.

Metric data provides quick feedback for software designers and managers. Analyzing and collecting the data can predict design quality. If appropriately used, it can lead to a significant reduction in costs of the overall implementation and improvements in quality of the final product.

References ;

- http://www.gita.state.az.us/policies_standards

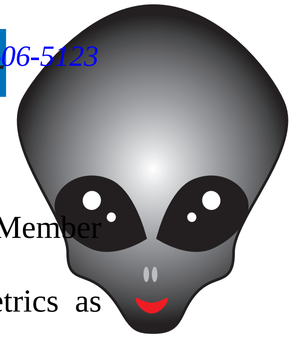
- Workflow Management Coalition, *The Workflow Reference Model*. 1994, URL http://www.aiai.ed.ac.uk/ WfMC/.

- Whitten, J.L., L.D. Bentley, and V.M. Barlow, *Systems Analysis and Design Methods*. 1994: Irwin.

- Kwan, M.M. and P.R. Balasubramanian. *Dynamic workflow management: A framework for modeling workflows*. in *HICSS-30*. 1997. Hawaii.

- Alkadi Ghassan, Carver Doris L.: Application of Metrics to Object-Oriented Designs,Proceedings of IEEE Aerospace Conference, Volume 4, pages 159 - 163, March 1998.

- Archer Clark, Stinson Michael: Object-Oriented Software Measures, Technical Report CMU/SEI-95-TR-002, ESC-TR-95-002, Software Engineering Institute, Carnegie Mellon University, April 1995.



- Victor R. Basili, Fellow, IEEE, Lionel C. Briand, Walcélio L. Melo, Member IEEE Computer Society: A Validation of Object-Oriented Design Metrics as Quality Indicators, IEEE Transactions on Software Engineering, Volume 22, Number 10, pages 751 - 761, October 1996.

- Lionel C. Briand, Sandro Morasca, Member, IEEE Computer Society, Victor R. Basili, Fellow, IEEE: Defining and Validating Measures for Object-Based High-Level Design, Volume 25, Number 5, pages 722 - 743, September/October 1999.

- Chidamber Shyam R., Kemerer Chris F.: A Metrics Suite for Object Oriented Design, IEEE Transactions on Software Engineering, Volume 20, Number 6, pages 476 - 493, June 1994.

- Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John: Design Patterns CD: Elements of Reusable Object-Oriented Software, Addison Wesley Longman, Inc.1998. Shuqin Li-Kokko: Code and Design Metrics for Object-Oriented Systems, Helsinki University of Technology, 9 pages, 2000.