A Development Methodology for a New Generation

by Grant W. Fletcher of The Interface Group, Incorporated, and
Kathleen A. Sachara of The Haley Corporation

Abstract of the Paper

The traditional methodology applied to system development has been eclipsed by the advanced software technology applied in the current generation of computer programming languages, thereby creating the need for these standards to be re-evaluated with regard to non-procedural programming languages and re-defined, where necessary, to provide an environment in which productive development and recognized standards for maintenance and auditability may co-exist.

A Development Methodology for a New Generation is an evaluation of the traditional development approach against the reality of programming in a non-procedural language. The observations and conclusions of this exercise form the basis of what is later presented as one proposed methodology in which the benefits of development in a non-procedural language and standards for maintenance and auditability are recognized as equally critical factors of successful system development.

The objective of the paper is to raise general awareness of the strengths and weaknesses of traditional development methodology with regard to current software development tools and languages.

Agenda of the Presentation

Traditional Methodology for System Development.

A review of the traditional standards for system development and the traditional project life cycle is presented, giving the audience a point of common reference for the discussion that follows.

Traditional Methodology Applied to a Non-Traditional Development.

A case study is presented in which a project is governed by traditional system development methodology and standards, but which involves code developed in a non-procedural language and the use of other software development tools.

Observations and Conclusions of the Exercise.

The strengths and weaknesses of the traditional approach to system development applied in a non-procedural language environment are discussed with reference to the preceding case study.

A Proposed Methodology.

The strengths of the traditional approach are augmented with revisions and additions to deal with the weaknesses of this approach found in the preceding case study. Specifically, the auditability and maintainability of the traditional methodology is enhanced with proposed standards for prototyping and code generation.

Traditional Methodology for System Development.

Development of computer based systems has evolved to a certain level of maturity with which potential users and data processing auditors are somewhat comfortable, but with the tools and techniques of such many data processing professionals find their creativity and productivity suppressed and frustrated. Traditional methodologies provide the user with reasonable certainties by which they may plan future staffing requirements and functions, and defineable project life cycles by which auditors may measure development responsiveness. A project life cycle built from a traditional system development approach provides comfort for the business manager and data processing auditor involved with a system implementation. The following is an example of one such project life cycle.

A Traditional Project Life Cycle

1. Project Initiation

2. Systems Analysis

3. Project Proposal

4. Project Specification

5. Development

6. Implementation

The first feature to notice about any traditional project life cycle is that each task is a defined step in the process of completing the project, and that all tasks, except the first, is contigent upon the completion of it's predecessor. The second important aspect of any traditional approach to system development is that project initiation originates from a user request for some function to be computerized, for some existing computer system to be enhanced, or for some existing computer system to be modified to resolve an error or problem.

1. Project Initiation. The catalyst for all system development activity under a traditional methodology is some form of request from a user, or potential user. A comprehensive methodology would require that the request be documented in some formal manner and logged to provide an audit trail. The user would be expected to provide a large amount of detail about their request. This may include a description of the function involved, how and why it is currently performed, who performs the procedures of the function, and what benefit they may expect from the implementation of the requested work. One may expect that this request would originate at a very low level on the corporate hierarchy and have to work it's way through several levels of supervisory personnel and management before it is actually submitted to systems development for consideration.

2. Systems Analysis. Since the project is initiated by a user, who should not be expected to fully understand the implications of their request throughout the organization as a whole, and who may not be aware of all available alternatives that may address the requirement that they have raised, a member of the systems development team will prepare an analysis of the request, as presented. A thorough systems analysis should entail a complete

description of the function involved and it's relationship to other elements of the organization, a compilation of all available alternatives with limited judgement on their individual merits and demerits, an analysis of the information required and generated by the function, an analysis of the procedures required or currently used to perform the function, a description of the hardware used or required to automate the task, and an evaluation of the future evolution and maintenance of any implemented system.

3. Project Proposal. The systems analysis produced above is used to form the basis of a project proposal which isolates the alternative from the systems analysis that best suits the requirements of the user and conforms to all relevant company policies and other procedures. The project proposal is written in language that is understandable by the users concerned with the implementation of the project, and describes information, procedures, process environment, implementation, and methodology for future enhancement relevant to the project. A project timetable should be used to provide the user with some idea of when staff functions will be affected by system implementation. The project proposal is either accepted or rejected by the user, supervisor, or manager because of it's understood appropriateness in response to the original request.

4. Project Specification. The project proposal accepted for development by the user community will not provide sufficient technical detail to remove all ambiguity for the programming staff. Therefore, information procedures, and environments may be re-written using schemas or pseudocode to better illustrate the software design and hardware configuration. Also, the user community is not directly concerned with the techniques that will be employed to test and debug, nor the procedures that will be used to implement the system under development. These tasks will not be described in the project proposal, but will be included in the formal project specification on which the development will be based.

5. Development. The development work occurs within traditional project life cycles begins once the users have been removed from the evolution of the request by accepting a given project proposal and a formal project specification has been drafted. The work of the development group is dictated by the wording of the project proposal, or the project specification, and not directly influenced by further user input or reaction to the work produced. Development management will often introduce and enforce various policies and formal procedures that will control how development work is accomplished. Typcially, one finds conventions for file naming, locality, and archiving that address data processing auditor concerns, as well as some standards that may dictate coding and testing of programmes that facilitate uniformity of installed work.

6. Implementation. The objective of any project life cycle is to install or implement a new or enhanced procedure or set of procedures. A traditional project life cycle completes itself when tested work is physically moved from the work environment, where all development work occurs, into the agreed environment where users of the completed system are given general access. Traditional methodologies that have been formalized will include defined procedures to accomplish the task of moving completed work and cleaning-up the work environment. Our traditional project life cycle dictates that discrepancies between the completed work and the actual user requirement, that

may have changed since the project proposal was accepted or have been poorly verbalized in the first place, must be addressed with a new project.

Traditional Methodologies Appled to a Non-Traditional Development.

Can the above project life cycle, or any similar traditional project life cycle be effectively applied in a situation where more advanced software tools are applied? The presentation of the paper will become an open forum to identify the tools that are used by the participants, and then follow a traditional project life cycle. The tools and techniques discussed will be applied in a case study manner.

Observations and Conclusion of the Exercise.

The purpose of the presentation of the paper will be to have those participating note observations and draw conclusions based upon the application of fourth generation tools and techniques within a traditional project life cycle.

A Proposed Methodology.

Traditional project life cycles tend to be the most weak in their responsiveness to evolving user requirements, although their primary purpose is to provide a high degree of certainty for the end user and associated auditors. To provide this certainty, a traditional project life cycle sacrifices the ability of the developers to address changes during the development stage of the project life cycle. Fourth generation software and other advanced programming tools and techniques exist to expedite the tasks involved with development work. The current trend in development tools, as evidenced by various prototyping tools and techniques is to not only expedite the project life cycle, but also questions the definition of a project with a clearly defined begin and end. Prototyping is not a methodology in itself, and so requires a mthodology that can recognize the abilities of prototypers to evolve systems with the direct input of an end-user, or group of users.

Implementing a prototyping tool while maintaining a traditional methodology restricts the potential of the tool. A traditional methodology requires a traditional specification. The requirement of a specification to direct a project is not a problem in itself, but the unchangeable nature of a specification does impede the reactiveness of prototype development. The existence of a specification in language that is separate from the prototype is definitely desired, in the same way that a specification that is separate from

the source code of any system is desired. It provides a clear declaration of the assumptions of the system designers. What must be reconciled, however, is the ability of a specification to evolve in the same manner that a prototyping tool will enable a system to evolve during it's development.

A traditional development methodology does provide a defined work flow by which productiveness and responsiveness may be measured. More important to the user is the defined timetable by which system implementation may be governed. The users of computer systems have their own staffing and procedural requirements that may have to be co-ordinated with the implementation of automated systems. End-user management also needs to be able to judge the effectiveness of procedures requested and subsequently impelemented in order to make intelligent decisions about future system requirements.

In order to satisfy the realities of programming in a business environment, it is also necessary to be able to deliver completed systems to end-users with which they may process their information with the certainty that that which produced correct results today will produce similar results tomorrow. Any methodology that is used by designers and developers of business systems has to recognize the reality that users cannot function effectively in a system environment that is constantly changing.

The problem in proposing a methodology that enhances today's development software is the reconciliation of the effectiveness and responsiveness that they offer with the necessity to deliver reliable business systems. With no defined development methodology, a project is assured to fail in both effectiveness and depandability.

A Proposed Project Life Cycle

1. Project Initiation

2. Systems Analysis

3. Initial Prototype

4. Systems Specification

5. Prototype Evolution

6. Implementation

A project must have a defined begin and end to provide a sufficient amount of certainty by which users may manage their expectations, staff training, and procedural changes. A defined begin and end also provides the ability to measure the responsiveness of system developers to the requirements of the users of their business systems. The catalyst for system development activity should remain the formal definition of the user's requirement, and the next logical step that follows should be the investigatory work of the systems analyst that assures the user that the designer of the resolution to their request truly understands the request and it's implications to the business as a whole. Project initiation and systems analysis are tasks that should be addressed in the same manner, regardless of whatever tools and techniques are applied to deliver the completed software.

After comprehensive systems analysis work has been completed, the analyst should be free to design a prototype that may be used as a discussion tool with all concerned users, rather than drafting a project proposal. In the same

manner that a project proposal is used to form the basis of a technical project specification, the initial prototype of a business system should be documented in such a manner that will clarify all things that have been assumed by the designer and the involved users. The initial prototype is evolved, but each time a new assumption is written into the system, the systems specification is evolved as well. The objective being to deliver a comprehensive prototype that may be given to the user community as a completed system, while maintaining a comprehensive explanation of the evolution that produced the prototype.

Project completion is still defined by the implementation of a prototype that is acceptable to the user as a working business system. User training becomes less of an issue as user involvement is present throughout the development, and the users are thus provided the opportunity to develop their procedures and manage staff training accordingly. Defined project completion also gives a point of reference to measure development effectiveness. An accepted prototype still acts in the same manner as an accepted project proposal in that it terminates one project and requires a new project life cycle to reconcile new requirements. However, the point in time at which the user freezes their requirement becomes sufficiently late in the project life cycle that immediate enhancement caused by new user procedures, policy, or functions is unlikely.