Rapid Prototyping as a Structured Methodology

Horst Lichter, ABB DECRC

1 Introduction

In view of the problems associated with software development and the criticism aimed at traditional life cycle plans, prototyping has, in recent years, increasingly become an approach that is adopted to improve the planning and success of software projects.

However, prototyping is often seen as an unstructured method used to rapidly develop a mockup of the desired application in a quick and dirty manner. But this is only one aspect of prototyping. In order to get all benefits from the prototyping approach, it is necessary to build several kinds of prototypes in a structured developement process.

In this contribution the basic terminology, the main concepts behind prototyping, the goals, and well known prototype construction techniques are introduced.

2 Prototyping Concepts

The term **Prototyping** has been used with various semantics in the literature ([2]). The first consequence of this apparent vagueness is that Òanything goes.Ó There is also, however, the implicit assumption that a method that cannot be defined exactly cannot serve as the conceptual basis for professional software development.

Floyd in [1] presented a first set of definitions of the term prototyping and its various meanings. These definitions have been refined and consolidated during the last few years. I want to present the essential definitions and the conclusions for software design.

- **Prototyping** is an *approach* based on an evolutionary *view* of software development, affecting the development process *as a whole*.
- **Prototyping** involves producing *early* working versions (ÒprototypesÓ) of the future application system and experimenting with them.

Prototyping provides a *communication basis* for discussions among all groups involved in the development process, especially between users and developers.

3 Kinds of Prototypes

Three major activities of the software development process can be influenced by the construction of prototypes: starting the project, analyzing the business needs, designing and constructing the software system. To illustrate the relationship between a prototype and these activities, the following *kinds of prototypes* may be distinguished:

- A **presentation prototype** supports the initiation of a software project. It is of major importance whenever an explicit contract is to be set up between a client and a software manufacturer. During acquisition, a presentation prototype is used to convince the client that the future application system is either feasible or that its user interface and handling is in line with user requirements.
- The term **prototype proper** is used to describe a provisional operational software system that is constructed parallel to the information system model. A prototype of this sort is generally designed to illustrate specific aspects of the user interface or part of the functionality and helps to clarify the problem in hand.
- A prototype that is designed chiefly to help clarify construction-related questions facing the development team is

called a **breadboard**. A breadboard is derived from the information system model or the software specification. This kind of prototype is also encountered in traditional software projects, although the experimental approach associated with it is seldom given explicit recognition. Users are generally excluded from the evaluation of breadboards. To this extent, the use of breadboards is a restricted form of prototyping.

- If a prototype is used not only for experimental testing of an idea or for Òillustrative purposes,Ó but is actually used in the application area itself as the core of the application system, it is known as a **pilot system**. After reaching a certain degree of sophistication, the prototype is implemented as a pilot system and enhanced in cycles.

4 Goals of Prototyping

In a software project various issues arise and can be answered by building prototypes. Here, the focus is on the major problems arising in typical *development situations*, problems which prototyping is designed to help solve. Following [1], we distinguish between several different goals of prototyping.

Exploratory Prototyping is used where the problem in hand is unclear. Initial ideas are used as a basis for clarifying user and management requirements with respect to the future system. Particular importance is attached to examining a variety of design options as not to restrict ideas prematurely to one specific approach. The developers gain insight into the application area and into the users' work tasks and the problems they face.

Experimental Prototyping focuses on the technical implementation of a development goal. Through experimentation the users are able to further specify their ideas about the type of computer support required. The developers, for their part, are provided with a basis for appraisal of the suitability and feasibility of a particular application system. An essential aspect here is the communication between users and developers on technical problems and questions about software ergonomics.

Evolutionary Prototyping is not merely used as a tool in the

context of a single development project; it is a continuous process for adapting an application system to rapidly changing organizational constraints. This means that software development is no longer seen as a self-contained project, but as a process continuously accompanying the application.

4 Prototype Construction Techniques

If software construction is seen as the design and implementation of many layers, ranging from the user interface to the base layer, the subdivision into horizontal and vertical prototyping begins to make sense .

In **horizontal prototyping** only specific layers of the system are built, for example the user interface layer with its forms and menus, or functional core layers, such as data base transactions.

In **vertical prototyping** a chosen part of the target system is implemented completely (Òdown through all layersÓ). This technique is appropriate where the system's functionality and implementation options are still open.

A further criterion for classifying prototypes is the relationship between the prototype and the application system. Here, many different views are taken:

- A prototype is part of the application system specification: The application system is built based on an accepted prototype. The prototype serves specification purposes only and is not used as a building block in the application system itself; it is a Òthrowaway.Ó
- Prototypes are enhanced to produce the application system: As the use of workstations and very high level languages for commercial applications become increasingly widespread, technical differences between prototypes and application systems are gradually disappearing.
- Prototypes serve to clarify problems only: One extreme view is to build prototypes merely for the purposes of acquiring knowledge, with no intention of building an application system, at least for the time being. This form of prototyping is frequently encountered in research and development

institutes and universities. One may even find this approach in commercial data processing, if, within a feasibility study, the fundamental decision has to be made as to whether a software system is to be developed at all.

References

[1] Floyd, C. (1984): *A Systematic Look at Prototyping.* In: Budde et al. (eds) Approaches to Prototyping, Springer-Verlag, pp. 105-122.

[2] Patton, B. (1983): *Prototyping - a nomenclature problem.* ACM SIGSOFT Software Engineering Notes, Vol. 8, No. 2, pp. 14-16.