Project Management and Design for Syndromic Surveillance Software

David Calvert

Computing and Information Science, University of Guelph, Guelph, ON, Canada

OBJECTIVE

Interest in syndromic surveillance through automated software systems is becoming more common and with this interest is an increase in small to medium sized software development projects. This paper discusses some of the common project management problems which occur when developing software in a community which does not have a long history of working in this area.

BACKGROUND

Management of software development projects involves a collection of well understood issues which are not often found in other project management areas. Identifying and managing these issues primarily requires that the manager is aware of the potential problems which can arise while developing software and what are the appropriate measures to control such problems.

PROBLEM AREAS

Areas which commonly cause problems in new software development projects fall into several categories:

- Difficulties defining requirements
- Design choices with far reaching consequences
- Developer ability
- Usability and user involvement

DISCUSSION OF PROBLEM AREAS

Defining requirements involves listing *what* tasks the system is intended to perform. It is most often overlooked because it appears obvious to those who instigated the project. It generally results in a system which doesn't perform the desired operations. Another common outcome of poor requirements definition is the continuous changing of requirements which leads to difficulties in completing the system. The cost associated with repairing errors in requirements definition is extremely high. Conscious decision making and prioritization are needed to develop a good set of requirements for a project.

Design choices made early in the development process can have far reaching consequences which limit future attempts to enhance the software. In an effort to make a system which works for a specific set of circumstances it is easy to overlook future operations which are likely to be desirable. Examples of this include not planning for expansion and supporting a restrictive data formats.

Managing a software project involves managing the developers who are creating the software. This requires regular interaction with the developers and an understanding that their abilities and motivations may be different from your own. Design choices should be made based upon their benefit to the goals for the project.

Usability of software is strongly related to the involvement of the end-users of the system, during its development. Without user feedback, the system will almost certainly not perform the desired tasks or will be difficult to use effectively. The system should be reviewed by the users during development. Their feedback should be used to refine the software. Feedback can be achieved through different methods including discussion or prototyping.

CONCLUSIONS

Although these points appear relatively obvious, experience has shown that they are commonly overlooked in new software projects. An awareness of these issues can mean the difference between a useful project and one which reaches completion but is unusable.

Further Information:
David Calvert, dave@cis.uoguelph.ca