The Open University

Some challenges facing software engineers developing software for scientists.

Judith Segal

Empirical Studies of Software Development Group,
Centre for Research in Computing
The Open University
UK MK7 6AA
j.a.segal@open.ac.uk

Contents

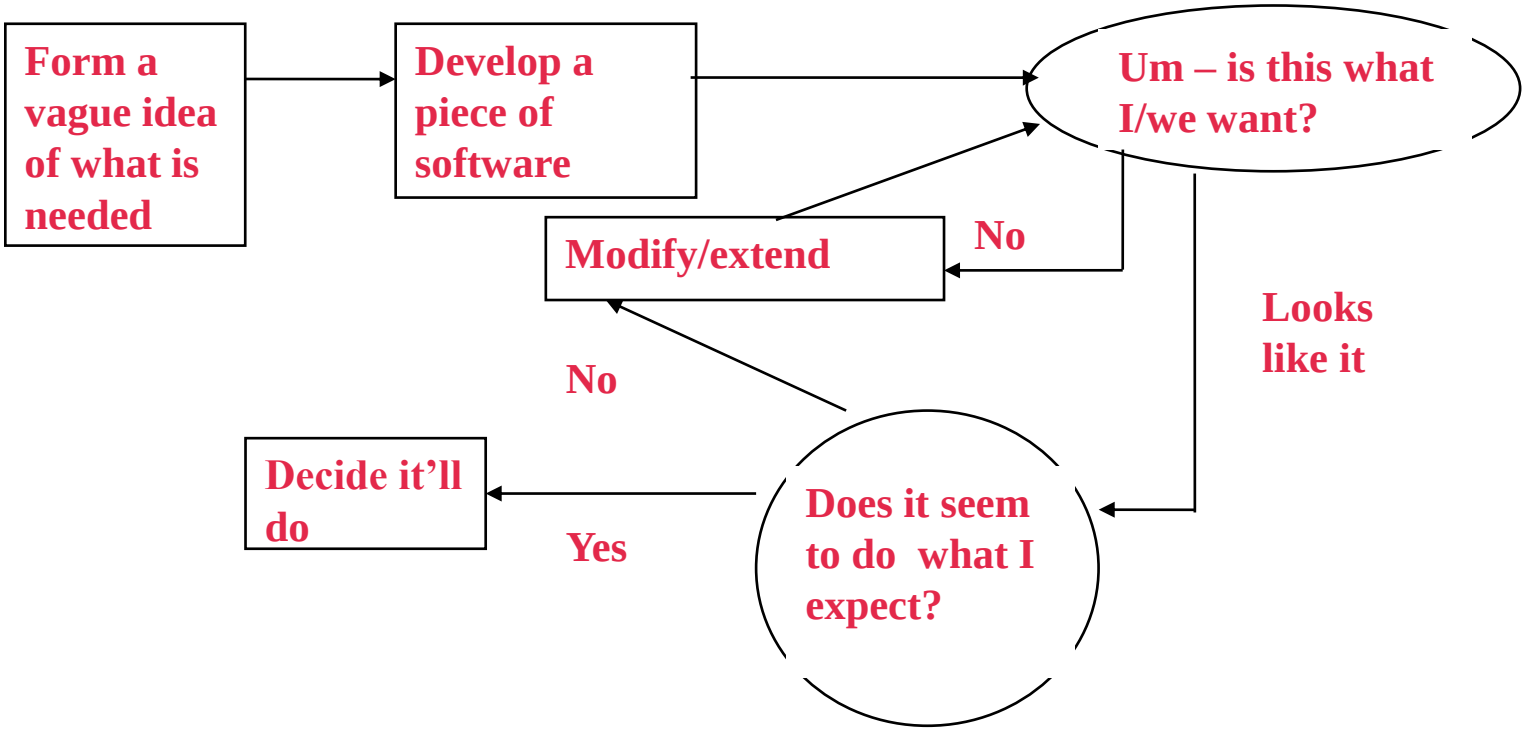
This list is far from exhaustive

- Based on my field studies (so not HPC)
- Socio-technical rather than technological

1. Those due to the scientists being ‘professional end-user developers’ (that is, having experience of developing their own software)
2. Those which are not unique to scientists but have particular salience in a scientific context
   a) The effective involvement of users
   b) Developing software for a community

1. Challenges that are reasonably well understood.

A model of scientists developing their own software – an iterative, incremental, feedback model



This is a very successful model – but only in a very **particular** context….

- The developers are the end-users or at least embedded in the end-user community. So:
  - Establishment of requirements
  - Testing

  are not considered to be major concerns.
- The software is developed to address a particular problem of a particular group at a particular point in time. So:
  - Comprehensibility, maintainability, portability not of concern

The success of this model (albeit in a very limited context) leads to the following risk:-

That scientists think they KNOW how to develop software in any context

The challenge to software engineers: managing scientists' expectations

- The establishing of requirements by software engineers is generally a more major resource intensive concern than scientists expect
- Ditto testing
- Software development in general takes longer than scientists expect.

This challenge is exacerbated by scientists' values:

- Scientists value scientific knowledge and skill over software development and skill
- "anybody can develop software"

2. Challenges which aren't well understood

- Engaging users effectively.

  Essential in the development of scientific software because of

  - the complexity of the domain
  - The preferred use of an iterative incremental feedback model.
- BUT scientists want to do science
- HOW can effective user engagement be enabled?

Developing scientific software for a community

As science “goes large” – development of cyberinfrastructure etc.

Some problems:-

- The culture of scientists is competitive
- “The tragedy of the commons”
- Variety e.g.of terminology (cf. problems with ontologies)

Thank you for listening

Any questions?