Please state your point of view on the issues of Mobile Software Engineering. Please specifically address the following 5 questions:

1. Your experience as a software engineer? Programming in multiple languages (micro-code, assembly, C, Fortran, LISP, Java, Python) since 1960; was on UML technical committee; ran reuse and software engineering projects at HP and U of Utah; director of HP software technology lab; wrote book and papers with Ivar Jacobson and others on Software Reuse, Software Engineering.

2. Your experience with mobile software development? Wrote Python and JAVA code for Nokia S60 series mobile phones; directed multiple student projects; ran a course on mobile application development.

3. How does traditional software engineering relate to the engineering of mobile applications and systems? In principle the SE process and techniques are same, especially for embedded systems; patterns, architecture and [reusable] components are different; UI design is significantly different; multiple system fragmentation encourages closer look at portable frameworks.

4. What are the distinguishing features of mobile software specification, architecture, development and testing that need special attention, skills, or innovation? The use of app stores as a primary distribution mechanism encourages rapid cycle development of smaller applications, with goal of viral adoption and “continuous” or frequent update. The importance of intermittent connection between device-based and cloud based services; UI design approach is similar, but design center is quite different from desktop/web apps- sensor rich, users whose attention might be elsewhere.

5. What is the suggested focus and agenda for mobile software engineering research and education?

   Education: Teach student that same SE principle should apply; teach patterns and best practices targeted to major mobile platforms, and interaction between device and cloud services. Teach both traditional compiled language development (cross develop) and rapid interpretive development; increased emphasis on testing.

   Research: How to design, build and test context-aware systems, integrating multiple hard- and soft- sensors information; also new “considerate system” design and UI models.