Large Scale Information Systems and Software

Victor R. Basili
Department of Computer Science
University of Maryland
College Park, Maryland

NIT Subcommittee of PCAST

Challenge

Software has become a critical part of any business, e.g., automotive, telecommunication, etc.

In the end, whoever leads in the development of software systems will lead in all aspects of business, product development, and military strength.

Challenge

Cutting edge software requires the building of

- **very large high quality software systems**
- **out of existing component parts**
- **by humans**
- **using combinations of techniques**
- **known to be appropriate and effective for the problem to be solved**

Competition

- Developing nations, (e.g., India, Eastern European nations),
  - learned what techniques most effective (cost, etc.)
  - under what conditions

- Developed nations (e.g., Germany)
  - fund applied software research and technology transfer
  - focus on empirical methods
  - observe/evaluate the effect of software techniques in practice

Areas of Research

(1) Improving the technology to assess and vet the effectiveness of various software techniques and technologies

(2) Developing techniques for building, evolving, and evaluating large systems from existing components

(3) Developing techniques for eliciting, specifying, and assuring the inclusion of non-functional requirements, such as safety, security, reliability, robustness, and availability into large software systems

(4) Developing repositories of knowledge gained from the above research activities that can evolve as new knowledge becomes available

Areas of Research

(1) Improving the technology to assess and vet the effectiveness of various software techniques and technologies

In the area of **software assurance** we talk about testing that a system has the capabilities we need, where do we talk about testing that the techniques provide the abilities we need to put the capabilities in place and evaluate their effects in different environments on different types of systems?

Software needs:

- Continual evaluation, assessment, and oversight
  - Continual improvement goals as technologies appear
  - Packaged knowledge in a usable form for practitioners: what works and under what conditions, e.g., a software engineering handbook,

Areas of Research

(1) Improving the technology to assess and vet the effectiveness of various software techniques and technologies

Implies: Studying the use of techniques and technologies in various contexts to understand how and when they work and make that knowledge available to organizations where software has become a critical part of the business, e.g., automotive, telecommunication, etc.

Implies: experimentation, testbeds, measurement, experience bases, results analysis and synthesis, applied research and technology transfer to test out techniques and technologies

Areas of Research

(2) Developing techniques for building, evolving, and evaluating large systems from existing components

We are building larger and larger systems

- no longer developed from scratch
- built from existing pieces of software, e.g., prior systems, COTS, open source components.
- built rapidly as requirements evolve
- Can’t longer wait 5 to 10 years for the system

Implies: Research in the areas of software component evaluation, acquisition, and integration; flexible architectures, with a sufficient focus on the idea that humans build these systems.

Areas of Research

(3) Developing techniques for eliciting, specifying, and assuring the inclusion of non-functional requirements, such as safety, security, reliability, robustness, and availability into large software systems

Dependability is in the eye of the beholder,

- There are many quality characteristics
- different stakeholders have different needs
- willing to pay different prices for the qualities they need.
- various qualities interfere with one another
- need to specify and test various levels of dependability
- need to make trade-offs among stakeholders and qualities.

How are these requirements elicited, formalized, traded-off, synthesized, and tested for?

Areas of Research

(3) Developing techniques for eliciting, specifying, and assuring the inclusion of non-functional requirements, such as safety, security, reliability, robustness, and availability into large software systems

Implies: Research in

- developing, applying, evaluating, evolving models of these qualities;
- ways to elicit system specific information to build the models;
- approaches to architect them into systems;.
- techniques to scale-up for complex system of systems:
- techniques to check that these qualities exist on various system components and aggregate them for the full system.

Areas of Research

(4) Developing repositories of knowledge gained from the above research activities that can evolve as new knowledge becomes available

Evaluations and experiments run on different systems, in different environments need to be combined

Study results, knowledge of when and where COTS products can be used, effective models for quality, need to be made available to

- researcher to analyze and synthesize the results and derive new knowledge
- practitioners, so they can know what to use and when

Implies: Meta-analysis, knowledge packaging approaches, …

Opportunities

Software is being developed all the time

We can
- Observe how it is being done: pitfalls, problems, …
- Build knowledge, supported by empirical evidence,
- Build testbeds at various levels to test, assess, and evolve
- Study software development in testbeds, real environments
- Learn what works and doesn't work
- Learn where new technologies are needed to fill the holes.
- Mature software techniques on testbeds
- Identify where new techniques need to be developed
- Lower the risk in their application

Recommendations

In summary, the NIT Research and Development Program should support:

1. experimentation with various (existing and to-be-developed) techniques and methods to understand their relationship with product qualities,

2. evaluations of various components (e.g., COTS) with respect to their quality and suitability for integration into different types of systems,

3. development of various approaches to elicit, model, and assure dependability requirements, and

4. build and maintain repositories that allow the analysis and synthesis of this knowledge

Impact

If we are the world leader in software development, we can

Choose what we want to build and outsource

Build the underpinnings of any technology

Lead in business, product development, and military strength.

Global Challenges/Opportunities

The challenge is to be able to plan, specify, build, and evaluate systems that are software intensive.

The opportunity is that the U.S. can be the best in such developments.

The focus should be on **applied research** – trying to better understand where are current approaches succeed, fall short, and can be improved

The empirical study component that is used in all other scientific disciplines needs to be applied to the software discipline.

Changes in the Federal government's investments in the NITRD Program

The SDP budget is quite small by comparison to the others and yet it is fundamental to all the others.

The software discipline is still immature

It is missing the experimental aspect of most scientific and engineering disciplines

Funding needs to be used to increase that aspect

To that end we need to fund the

- building and maintenance of testbeds
- Running of experiments
- Building and maintaining of the results of experiments so researches can analyze the results of different studies and create a more informed assessment of these techniques

Changes in the Federal government's investments in the NITRD Program

The applied research suggested here needs to be performed across all agencies, as the opportunities, problems, type of system, etc. are different in all agencies.

There should be some central agency recommending studies and integrating the obtained knowledge. This knowledge needs to be analyzed and synthesized to evolve the software discipline.

Changes outside the NITRD Program

More direct links with organizations developing software so they can be studied and be the benefactors of technology transfer as part of the package.

The organization provides testbeds but also gains from the identification and tailoring of the research results.

A good model is the Fraunhofer Centers in Germany who are funded to do applied research and technology transfer.

Funding consists of government supplied base funding as well as financial support from the organizations who are the benefactors of the transferred research to practice.

References

V. Basili, “Evolving and Packaging Reading Technologies”, *Journal of Systems and Software*, vol. 38 (1): 3-12, July 1997

V. Basili, F. Shull, and F. Lanubile, “Building Knowledge through Families of Experiments,” *IEEE Transactions on Software Engineering*, vol. 25(4): 456-473, July 1999

P. Donzelli and V. Basili, “A Practical Framework for Eliciting and Modeling System Dependability Requirements: Experience for the NASA High Dependability Computing Project,” *Journal of Systems and Software*, vol. 79(1): 107-119, January 2006

A. Endres, H. D. Rombach, A Handbook of Systems and Software Engineering, Addison Wesley, 2003.