Software Engineering is Multi-Disciplinary: Impact on Design for Change

W. Morven Gentleman

Dalhousie University

Morven.Gentleman@dal.ca

Software Engineering is

- the collection of practices and theory, tools and techniques that make the development and support of software a viable business
  - Software design is part of Software Engineering, but only a small part

SE is multi-disciplinary

- The *what* of SE: any software product has a domain of application. Most disciplines, indeed almost every domain of human endeavour, need supporting software
- The *how* of SE: many disciplines contribute to the effective production and support of software

Typical issues for What

- Domain specific science and technology
- Domain specific resources
- Domain specific standards
- Domain regulatory requirements
- Culture of the domain

Typical issues for how

- Reduced development cost
- Earlier time-to-market
- Higher quality
- Better predictability of product and process
- Broader applicability, amortization over larger marketplace

Examples

- Cognitive psychology, HCI and tool adoption
- Sociology, CSCW and support of long-lived systems
- Management science, software process, and strategies for fixed price contracts
- Statistics, test plans and performance tuning

Types of change

- Change in the marketplace
- Change in the customer organization
- Change in the supplier organization
- Change in the product

Change in the marketplace

- Legal compliance
- Background user experience
- Interoperability
- Competitors and partners
- Downsizing, outsourcing, virtual enterprise and the rise of SME
- Dehumanization of business interaction

Change in the customer organization

- Rollout: change in business processes
- Growth and retrenchment
- Centralization and decentralization
- Mergers: not all change is improvement
- Continuous upgrades: change in context

Change in the supplier organization

- Hand-off from development to maintenance (and back)
- Staff rollover: immigration and emigration
- Work structure change: process and organization
- Cost structure change
- Transfer of product support to a different supplier

Change in the product

- By definition, successful products are long-lived
- Over that time, changes must be accommodated
  - Inscale and mission
  - In economics
  - In platform
  - In available technology
  - In user expectations

Conclusion

- The software engineer must not only be adept in, and keep up-to-date with, the rapidly changing technology in the core of the discipline, but must maintain awareness of current relevant technology in related disciplines