

# Engineering Web Apps

CMPT 470 Guest Lecture

by  
Kathy Peters

© Kathleen Peters 2000-2001

1

## Overview

- Development difficulties/issues
- What is “software engineering”?
- Some “Good Practices” to improve efficiency and chances for success

© Kathleen Peters 2000-2001

2

## Developing any significant software application isn't easy

- Software systems/products are becoming larger and more complex, while becoming more vital to the success of the organizations that require them.

---

- average project likely to be 6-12 months behind schedule and 50-100% over budget

---

- significant #'s of projects fail completely; 2 out of 6 large projects are eventually canceled.

---

- for those projects that complete, significant #'s have deliverables with major quality problems; 75% of large projects are operational failures.

---

- Development, maintenance (and operational) costs are high and are often under-estimated.

© Kathleen Peters 2000-2001

3

## Is web development something new or just more of the same?

2 views, still being debated ....

### ◆ No

swr development is swr development; "web" just adds some new wrinkles (but what new technology hasn't); there's nothing here that hasn't been seen before

### ◆ Yes

we are discovering, inventing, building in a new frontier and our approach must be new to be successful; the "old ways" will only hold us back.

© Kathleen Peters 2000-2001

4

## Some of today's Web App Development Difficulties

- **Schedule Compression**  
enormous pressure to deliver quickly - in weeks, instead of months or years.
- **Technology Instability (a moving target)**  
very rapid technology change - new products, tools and techniques announced daily; lots of limitations and "bugs" in early versions
- **Requirements Instability (a moving target)**  
what are we building today? Tomorrow it will change or become obsolete!

© Kathleen Peters 2000-2001

5

## Web App Difficulties cont.....

- **Team Composition**  
multi-disciplinary - project is more like a film production; often geographically distributed
- **Staffing Shortage (it's a crisis!)**  
there not enough people with enough experience in the required technologies who have enough experience developing software. Need "immediate" productivity.
- **Vast and Variable User Community**  
millions of anonymous customers with varying expectations and skill sets; information derived through site statistics not direct communication

© Kathleen Peters 2000-2001

6

## Web App Difficulties cont...

- **Limitations and Vulnerabilities**

attempting to build complex UI on a (currently) limited framework;  
huge security concerns (public network; flawed 3rd-party software);  
is it a single "site" or a "product" - if the latter, more complications.

- **Testing (how much before and after you "go live"?)**

does it function? is it secure? is performance sufficient? Is it usable? does it "sell" (initial & return visits)? Are the perceived benefits being realized?

- **Continuous Enhancement**

a web site is NEVER done! New content & functionality may appear daily! Can this be properly prioritized and controlled?

## "Software Engineering" can help

- A disciplined approach to swr development (at the individual, team and organizational level) - not chaos!
- Consistent application of proven practices - others have learned what works and what doesn't so learn from them!
- "Just enough" process orientation (defining and documenting the procedures you will follow and what the deliverables will be) so that you can estimate the work to be done and you have a "road map" to follow - some level of predictability, less rework and "thrashing".
- The desire to improve, to observe and repair what fails

## But there is no “silver bullet”!

- Software Engineering won't make the problems and the risks go away - nothing can!
- Software Engineering can help you plan, monitor, control (to some extent), and cope with a very challenging and dynamic area of endeavor; you can make more informed decisions, you can suffer less “pain”, you can produce better quality software and better manage customer expectations.

## Some important “Good Practices” for developing Web Apps

- Project Management
- Risk Management
- Change Management
- 3rd-party Management
- Development Lifecycle & Process choices
- Test Planning and Test Cases

And there are lots more “good practices” than these to learn and use!

## Project Management

- Single biggest reason web projects fail is lack of proper management
- Few experienced project managers available (or appreciated!) - get one, or train one - don't start a web project without one!
- Mgr helps plan & monitors project; helps solve problems; provide leadership, not dictatorship; motivates, not threatens; coordinates all the different groups who are working to produce the result; helps manage customer expectations

## Risk Management

- “leading edge” or “bleeding edge” development is the most risky kind, and is the least predicable
- you **MUST** know what your highest risks are (most likely to occur, highest negative impact if they do)
- you **MUST** try to manage them (look for ways to reduce the risk, or reduce the impact if they occur)
- you **MUST NOT** give customers a single-date estimate for completion too soon! The “end” is a range that slowly narrows as you move forward.
- Fixed price contracts are risky!

## Change Management

- Change is a fact of life - deal with it!
- As things change,
  - ◆ Create a change request (CR)
  - ◆ Analyze the impact of the CR
  - ◆ Accept, reject the CR
  - ◆ Schedule the CR
  - ◆ Make the change(s) and inform everyone
  - ◆ Close the CR
- Use a tool to manage CRs

## 3rd-Party Management

- You have a “dependency” on any third-party involved in the web project - what happens when one or more don't deliver what you need when you need it? e.g.,
  - ◆ multimedia (graphics, sound, video) designers
  - ◆ content writers/editors
  - ◆ 3rd-party vendors (tools, swr components, hardware, hosting services, shippers, ...)
  - ◆ contractors and consultants
- Need to make the 3rd-party requirements explicit (clearly define everything) and monitor progress/use - don't let problems blindside you.

## Lifecycle; Development Process

- An “iterative” lifecycle, with a series of releases, is the best approach; must do some prototyping
- “light methodologies” are being proposed, e.g.,
  - ◆ eXtreme Programming
  - ◆ Scrum
  - ◆ Adaptive Software Development
- “traditional” process can be “lightened up” too (SPC’s eSETweb product does this)

## Test Planning & Test Cases

- Start planning “system testing” early - as soon as you have some requirements you can start
- Document your test plan and test cases; prioritize - testing phase always ends up tight so know what you can cut
- How much testing is enough? ... “it depends”! How much risk are you prepared to accept?
- You don’t have to do it all yourselves - there are organizations that specialize in testing! (QA Labs in Vancouver is one)

## In conclusion ...

- There are all sorts of interesting technical challenges in building web applications, and there's lots of demand for people who can help build them
- However,
  - ◆ technical skills are only a small part of what you should know to be a valuable member of a web development team. Expand your horizons; be curious and interested in all sorts of related areas.
  - ◆ It isn't a stress-free environment or a perfect world where you get to work on just the "fun stuff" - it can be frustrating and exhausting too. Pace yourselves as there will be plenty of opportunity for the foreseeable future.

## More information ?

I'll email a list of references to your professor this week ....