In July 2004, Microsoft working on Longhorn, successor to Windows XP. Jim Allchin, senior executive, claims that Longhorn will never work properly. Reason: software too complex to be developed using synchronize-and-stabilize life-cycle model. Conclusion: need to start over with new model.

Ideal Microsoft Development Life-Cycle

- Produce a new version of Windows every few years
  - Bug free as possible
- Frequent updates between versions
  - Add enhancements
  - Match competitor features

Synchronize-and-Stabilize Model

- Requirements analysis — interview potential customers
  - E.g., Bill Gates wanted to add WinFS to Windows
- Draw up specifications
- Divide project into 3 or 4 builds
- Each build is carried out by small teams working in parallel

Synchronize-and-Stabilize Model (2)

- At the end of the day — synchronize (test and debug)
- At the end of the build — stabilize (freeze the build)
- Supposedly, components always work together

Reality of Longhorn Project

- 4000 independent programmers
- 8'x11' map of how Windows’ pieces fit together looked like a haphazard train map of crossing tracks
- Troubleshooters had to manually search through thousands of lines of code to find a bug
- Daily synchronization impossible

Security Issues

- In addition, Windows architecture made it an easy target for viruses
- Fixing security issues diverted top programmers from Longhorn
### Economic Issues
- Competitors produce new software features faster than Microsoft
  - Google Gmail, Mozilla Firefox
- Best programs like Lego blocks
  - Each performs a single function
  - Designed to be connected to larger whole
  - Can be added incrementally

### Players
- Jim Allchin: Ph.D. in computer science, background in large corporate systems design, head of division
- Brian Valentine: one of the best-known "shipper", people known for turning around troubled projects, co-proposer and enforcer
- Amitabh Srivastava: computer science purist, newcomer to Windows, co-proposer and tool builder
- Dave Cutler: revered, somewhat reclusive, software guru, proposal evangelist

### Proposed Solution
- Restart Longhorn using Windows 2000 (corporate server version) as fresh base
- Design features to allow be easily plugged in or pulled out
  - Initial release to be simpler
- Use tools to keep code base clean
  - Automatically reject bug-laden code

### Automated Test Tools
- Automate testing that had been done by hand
- If a feature had too many bugs, rejected from being used in Longhorn
- If a programmer has too many outstanding bugs, banned from writing new code
- Goal is to get programmers to “do it right the first time”

### Reactions
- Bill Gates
  - Take more time to finish Longhorn as is
  - Is this really the right thing to do?
  - Automated tools unproven; could take time away from devising innovations
- Programmers
  - Extra bureaucracy
  - Stifles creativity

### How to get programmers to buy in?
- Give tools to revered guru among Windows programmers, ask for his support
- Have him give a presentation in which he claims to have used the tools to produce nearly bug-free Windows code
Acceptance of Proposal

- August 2004
- Restart Longhorn Project using code base from Windows 2000
- Push back delivery of Longhorn by at least a year (second half 2006)
- Reduce features
  - For example, WinFS not in initial release

Results

- As testing tools were refined and programmers cooperated, quality of code improved
- Time to synchronize fell to a few days
- Missed June 2005 deadline for first beta release, but did so in July
  - Couple thousands of problem reports instead of tens of thousands

Results (2)

- Test versions with new features released monthly
- Microsoft Office Group using testing tools to improve its code

Conclusion

- Incentive to change the way things are done is economic
- A modular approach to developing software
  - Reduces bugs, reducing corrective maintenance,
  - And allows perfective and adaptive maintenance to be performed more rapidly

Reference