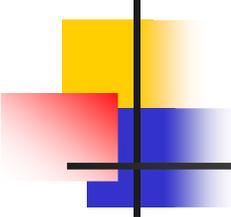


Verification Predictability

Nir Weintroub

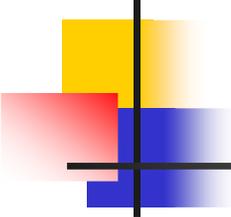
Noam Elbaum





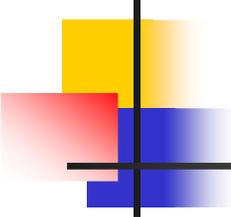
Outline

- Major problems for predictability
- Stages for predicting verification progress
 - Initial time/effort estimation for project
 - During project
 - Track verification progress
 - How much to the end?
- Methods for progress tracking
- So ... why doesn't it work as planned?



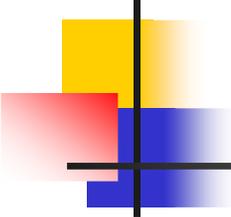
Major problems for predictability

- High dependency on other non controllable components
- Verification is not always proportional to design complexity/size and its importance
- In cases there is only serial progress
 - Non corrected bug might shadow others
- Quality measures are not available



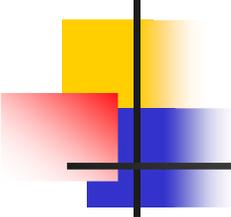
Initial time/effort project estimation

- Estimated resources needed
 - Highest priority
 - Time
 - Manpower
 - Highest priority
 - Computing resources
 - Tools and licenses
- Available information ?
 - Deadline
 - Spec



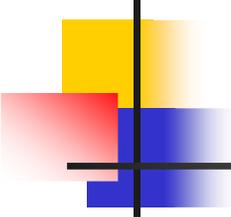
Initial time/effort project estimation – 2

- Estimation method
 - Estimation on previous projects
 - Are the projects similar (complexity)??
 - Out-of-the-blue
 - Manager experience
 - Gut feeling



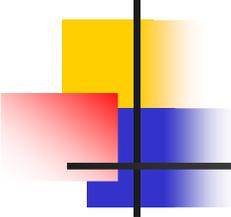
Initial time/effort project estimation – 3

- What parameters can be used for reducing errors?
 - Can we estimate complexity of design/verification?
 - Quality of designers <-> verification engineers
- How much do we give to different project stages
 - Environment building (probably easier to estimate)
 - Feature/block level
 - Integration



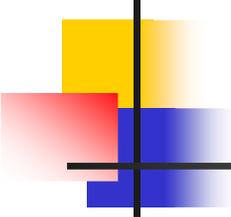
During project phase

- Can the dependencies on design stability can be controlled??
- Can other dependencies be controlled
 - Computer/tools Resources?



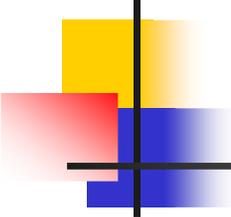
During project phase - 2

- Problematic stages
 - Verification + design ramp-up
 - Hard to predict effort/time
 - Finalizing verification estimation roadblock
 - Work quality – verification and design
 - Complexity of design



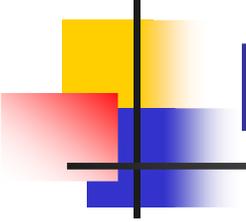
During project phase - 3

- How to identify delays?
 - During implementation stages (Simple) there should be a defined schedule per task.
 - During coverage collection (More complicated)
 - there should be a defined coverage collection plan (Percent or Features ?)
 - Bugs per week – lots of bug found in design indicate that there is lot to do, delay can be identified by a steady (and high) bug rate
 - Engineers inputs
 - Do we need a formal status check?
 - In what stages of the project?



Methods for progress tracking

- Bug track – quantity and quality
- Functional coverage – CDV
- Code coverage
- Test Plan lists (separate or not from functional coverage)
- Additional issues
 - Weights for features
 - Combine all above



So ... why doesn't it work as planned?

- Maybe next time it will ??