Globally Distributed Software Development Project Performance: An Empirical Analysis

Narayan Ramasubbu and Rajesh Krishna Balan
Singapore Management University

Roman Saratz
Main Questions

- **Dispersion Impact:** To what extent does dispersion affect software productivity and quality?

- **Impact Mitigation:** To what extent can investments in structured processes mitigate the effect of dispersion?

- **Relative Effects:** What are the relative effects of individual PM-practices in improving distributed project performance?
Research Model

- Work Dispersion
- Quality Management Approaches
- Project Performance
- Control Variables
Work Dispersion

Work Dispersion = $100^2 - 
\left(\%\ \text{effort at 1}\text{st development center}\right)^2 - 
\left(\%\ \text{effort at 2}\text{nd development center}\right)^2$
Quality Management Approaches

- **Prevention-based approach**: Avoid the occurrence of errors by additional training

- **Appraisal-based approach**: Proactively assess progress, performance and quality of intermediate artifacts

- **Failure-based approach**: Checking specification compliance and subsequent defect correction activities
Project Performance

- Development Productivity: Ratio of software code in function points to the total development effort in person-hours.

- Conformance Quality: # of unique problems reported by customers during acceptance tests and production trials
  Conformance Quality = 1 / (defects + 1)
Control Variables

- Productivity Variables
  - Team Size
  - Reuse

- Quality Variables
  - Code Size
  - Upfront Investment

- Common Variable
  - Design Rework
Empirical Equations

Based on the research model:

\[
\ln(\text{development productivity}) = \\
\alpha_0 + \alpha_1 \ln(\text{conformance quality}) + \alpha_2 \ln(\text{work dispersion}) + \\
\alpha_3 \ln(\text{prevention}) + \alpha_4 \ln(\text{appraisal}) + \alpha_5 \ln(\text{failure}) + \\
\alpha_6 \ln(\text{reuse}) + \alpha_7 \ln(\text{design rework}) + \alpha_8 \ln(\text{team size}) + \varepsilon_1
\]

\[
\ln(\text{conformance quality}) = \\
\beta_0 + \beta_1 \ln(\text{development productivity}) + \beta_2 \ln(\text{work dispersion}) + \\
\beta_3 \ln(\text{prevention}) + \beta_4 \ln(\text{appraisal}) + \\
\beta_5 \ln(\text{failure}) + \beta_6 \ln(\text{design rework}) + \beta_7 \ln(\text{code size}) + \\
\beta_8 \ln(\text{upfront investment}) + \varepsilon_2
\]
Data Collection

- One Leading Software Company
  - Over 19’000 people in 17 countries
  - Revenue of more than one billion dollars

- Highly adopted to global delivery model for its services

- Information gathered on 42 completed projects in a two year time period

- Additional structured interviews with multiple employees
Results

- Work dispersion negatively affects development productivity (exponentially)

- Dispersion has only indirect effect on conformance quality in high process maturity environments
Result: Quality Management Approaches

- **Productivity:**
  - Highest Impact from appraisal-bases approach
  - Followed by failure-based approach
  - Prevention statistically not significant

- **Conformance Quality:**
  - Highest impact from failure-based approach
  - Followed by prevention-based approach
  - Appraisal statistically not significant
Main Questions Answered

- **Dispersion Impact**: Dispersion has a significant effect on productivity and a harder-to-capture secondary effect on quality.

- **Impact Mitigation**: Effects can be significantly mitigated through use of structured software engineering processes.

- **Relative Effects**: different management approaches have significantly different impacts on different dimensions of project performance.
Results Discussion

- Difficulties of managing uncertainties caused by interdependent tasks
  - Information asymmetry between remote teams
  - Ambiguous authority

- Preventive and appraisal-based approaches facilitate learning-before-doing, failure-based approach facilitates learning-by-doing

- Learning-by-doing might be the most effective learning method in distributed development environments
Limitations

- Already observed in high process maturity environment
- Only a one to two year period
- Model did not consider task level interdependencies among individual team members
- Only custom business application software development projects
- Only one firm
Conclusion

- Significant effect on productivity and indirectly on quality
- Those effects can be mitigated
- High quality software processes help overcome dispersion
Questions
Bonus Slides
Function Points

- A function point is a unit of measurement to express the amount of business functionality an information system provides to a user.

- The functional user requirements of the software are identified and each one is categorized into one of five types: outputs, inquiries, inputs, internal files, and external interfaces.

- Once the function is identified and categorized into a type, it is then assessed for complexity and assigned a number of function points. Each of these functional user requirements maps to an end-user business function, such as a data entry for an Input or a user query for an Inquiry.

(Source http://en.wikipedia.org/wiki/Function_point_analysis)
Future Work

- Determine how to reduce interdependence between tasks in distributed environments
- Understand how the results apply to other type of companies
- Develop a general model that applies across different types of software methodologies
Results
Results