A Unifying Theoretical Foundation
(or perhaps better: Framework)
for
Software Engineering

Dewayne E Perry
Center for
Advanced Research in Software Engineering (ARiSE)
perry@mail.utexas.edu
Separation of Concerns

- An important separation of concerns - distinguish between
  - Theories about *software engineers*
    - As people (individual or in teams), as designers, as creators, as programmers, as architects, as engineers, etc
    - How people and teams interact, cooperate to create and evolve software systems
    - Cognition is located here
  - Theories about *software engineering*
    - The actual crafting and engineering of software systems
    - The structure of the artifacts
    - How to create and evolve them
    - Techniques and structures to manage complexity is here
  - Theories about *software project management*
    - Managing software engineers and software engineering
    - How to best organize and assign people given resources
    - Managing project resources, roles, etc
Separation of Concerns

★ Theories about the relationship between the theories of software engineers and software engineering
  ➢ Eg, various cognitive issues for SEs are related to various principles and structures used in SEing

★ Theories about the relationships between theories of project management, software engineers, and software engineering
  ➢ Eg, SPM is concerned about the utility and effectiveness of SEs and the progress, quality and cost of SEing
  ➢ Eg, PM metrics and productivity of SEs
  ➢ Eg, SE roles and responsibilities wrt SEing artifacts

★ I am primarily interested in Theories about Software Engineering
★ But ultimately will want to compose?/integrate? theories of SE, SPM, SEing, SE-SEing, and SPM-SE=SEing
Adolph/Kruchten Theory - P

- "A Grounded Theory is a set of integrated conceptual hypotheses systematically generated to produce a theory"
- "Grounded Theory generates a substantive theory that explains participants' behavior as a set of integrated hypotheses"
- "the main concern of people involved in the process of software development is getting the job done and that different points of view and expectations create impediments - a perspective mismatch"
- When a perspective mismatch is discovered, people converge their mismatched perspectives by reaching out and negotiating a consensual perspective (which I refer to as observations - or for grounded theory, hypotheses)
Adolph/Kruchten Theory - P

- Incorporates D and E
- New elements for P
  - P person (ie software engineer)
  - O observations - negotiated perspectives - hypotheses
  - R researcher (a special subset of P)
  - T_{se} theory of software engineering

- At an abstract level
  - P+ * D \rightarrow O+
    - One or more people derive one or more observations about creating/evolving a design
  - P+ * (E:D) \rightarrow O+
    - One or more people derive one or more observations about evaluating a design
  - R+ * O+ \rightarrow T_{se} or alternatively R+ * O+ * T_{se} \rightarrow T_{se}
    - One or more researchers create or modify a theory of SE using the observations
Adolph/Kruchten - P

- Need to expand D
  - \( P^+ \ast W \to O^+ \)
  - \( P^+ \ast T \to O^+ \)
  - \( P^+ \ast M \to O^+ \)
  - \( P^+ \ast (W \to T) \to O^+ \)
  - \( P^+ \ast (T \to M) \to O^+ \)
  - \( P^+ \ast (M \ast W \to T) \to O^+ \)
  - \( R^+ \ast O^+ \ast T_{se} \to T_{se} \)
  - \( P^+ \ast P \to O^+ \)
  - \( P^+ \ast O \to O^+ \)

- Need to expand E:D
  - \( P^+ \ast E:W \to O^+ \)
  - \( P^+ \ast E:T \to O^+ \)
  - \( P^+ \ast E:M \to O^+ \)
  - \( P^+ \ast E:(W \to T) \to O^+ \)
  - \( P^+ \ast E:(T \to M) \to O^+ \)
  - \( P^+ \ast E:(M \ast W \to T) \to O^+ \)
  - \( R^+ \ast O^+ \ast T_{se} \to T_{se} \)
  - \( P^+ \ast E:P \to O^+ \)
  - \( P^+ \ast E:O \to O^+ \)
Adolph/Kruchten - P

- \( P \ast E : (T \rightarrow M) \rightarrow O^+ \) - Model E from Atomic to Open Structured
  - \( P^+ \ast W : (T \rightarrow M) \rightarrow O^+ \)
    - People’s observations about the world of creating a model from a theory
  - \( P^+ \ast T : (T \rightarrow M) \rightarrow O^+ \)
    - People’s observations about a theory of creating a model from a theory
  - \( P^+ \ast H : (T \rightarrow M) \rightarrow O^+ \)
    - People’s observations about an hypothesis about creating a model from a theory
  - \( P^+ \ast R : (T \rightarrow M) \rightarrow O^+ \)
    - People’s observations about a regimen about creating a model from a theory
Adolph/Kruchten - P

★ $P^+ \ast ((W \rightarrow T):(T \rightarrow M)) \rightarrow O^+$
  $= P^+ \ast W:((T \rightarrow M) \rightarrow T:(T \rightarrow M)) \rightarrow O^+$

  People's observations about deriving a theory of creating a model from a theory, from a world of creating models from theories

★ $P^+ \ast ((T \rightarrow H):(T \rightarrow M)) \rightarrow O^+$
  $= P^+ \ast T:((T \rightarrow M) \rightarrow H:(T \rightarrow M)) \rightarrow O^+$

  People's observations about deriving an hypothesis about creating a model from a theory, from a theory of creating models from theories

★ $P^+ \ast ((H \rightarrow R):(T \rightarrow M)) \rightarrow O^+$
  $= P^+ \ast H:((T \rightarrow M) \rightarrow R:(T \rightarrow M)) \rightarrow O^+$

  People's observations about deriving a regimen for evaluating the derivation of an model from a theory, from an hypothesis about creating models from theories

★ $P^+ \ast ((R^*W \rightarrow T):(T \rightarrow M)) \rightarrow O^+$
  $= P^+ \ast (R:(T \rightarrow M) \ast W:(T \rightarrow M) \rightarrow T:(T \rightarrow M)) \rightarrow O^+$

  People's observations about reconciling the evaluation of a theory of creating a model from a theory, with the world of creating models from theories, possibly modifying that evaluated theory
Adolph/Kruchten - Model E:P

To evaluate the creation/evolution of P

- $E: (P^+ \ast W \rightarrow O^+)$
- $E: (P^+ \ast T \rightarrow O^+)$
- $E: (P^+ \ast M \rightarrow O^+)$
- $E: (P^+ \ast (W \rightarrow T) \rightarrow O^+)$
- $E: (P^+ \ast (T \rightarrow M) \rightarrow O^+)$
- $E: (P^+ \ast (M \ast W \rightarrow T) \rightarrow O^+)$
- $E: (P^+ \ast E: (W \rightarrow O^+))$
- $E: (P^+ \ast E: (T \rightarrow O^+))$
- $E: (P^+ \ast E: (M \rightarrow O^+))$
- $E: (P^+ \ast E: (W \rightarrow T) \rightarrow O^+)$
- $E: (P^+ \ast E: (T \rightarrow M) \rightarrow O^+)$
- $E: (P^+ \ast E: (M \ast W \rightarrow T) \rightarrow O^+)$
- $E: (R^+ \ast O^+ \ast T_{se} \rightarrow T_{se})$
Adolph/Kruchten - Model E:P

- **E:** (P+ * (T → M) → O+)
  - **W:** (P+ * (T → M) → O+)
    - A world of peoples observations about deriving a model from a theory
  - **T:** (P+ * (T → M) → O+)
    - A theory about people’s observations about deriving a model from a theory
  - **H:** (P+ * (T → M) → O+)
    - An hypothesis about people’s observations about deriving a model from a theory
  - **R:** (P+ * (T → M) → O+)
    - A regimen for evaluating people’s observations about deriving a model from a theory
    - Deriving a theory about peoples observations about deriving a model from a theory from the world of peoples observations about deriving a model from a theory
Adolph/Kruchten - Model E: P

★ (T → H): (P+ * (T → M) → O+)
    = T: (P+ * (T → M) → O+) → H: (P+ * (T → M) → O+)

- Deriving an hypothesis about peoples observations about deriving a model from a theory from a theory of peoples observations about deriving a model from a theory

★ (H → R): (P+ * (T → M) → O+)
    = H: (P+ * (T → M) → O+) → R: (P+ * (T → M) → O+)

- Deriving an hypothesis about peoples observations about deriving a model from a theory from a theory of peoples observations about deriving a model from a theory

★ (R*W → T): (P+ * (T → M) → O+)
    = (R: (P+ * (T → M) → O+) * W: (P+ * (T → M) → O+)) → T: (P+ * (T → M) → O+)

- Reconciling the results of a regimen evaluating peoples observations about deriving a model from a theory, with the world of peoples observations about deriving a model from a theory, possibly modifying the evaluated theory
Batory Theory of Design F

- “Feature Oriented Programming (FOP) is a design methodology and tools for program synthesis. The goal is to specify a target program in terms of the features that it offers, and to synthesize an efficient program that meets these specifications”
  - “the constants and functions of a domain model — which is an algebra — can be implemented with many different technologies”
  - “equational representations of programs are very powerful”
  - “Design rules capture semantic constraints that govern legal compositions”
Batory Theory of Design F - Model

- Elements in F (simplified - ie no iteration)
  - W  world
  - T  theory
  - F  feature
  - A  algebra
  - R  design rule
  - M  model
  - W → T  derive a theory from the world
  - T → F+  derive features from the theory
  - A * F+ * R+ → M  derive a model from the features via the algebra
  - M * W → W  inject the model into the world
E:F - Evaluating Design Theory F

- Evaluating F - E:F
  - E:W  evaluate the relevant world
  - E:T  evaluate the theory
  - E:F  evaluate the features
  - E:A  evaluate the algebra
  - E:R  evaluate the design rules
  - E:M  evaluate the model
  - E:(W \rightarrow T)  evaluate the process of deriving a theory from the world
  - E:(T \rightarrow F+)  evaluate the process of deriving features from the theory
  - E:(A * F+ * R+ \rightarrow M)  evaluate the creation of a model from applying the algebra and design rules to the features
  - E:(M * W \rightarrow W)  evaluate injecting the model into the world
Theory of Research D:F and D:(E:F)

- D:F
  - D:W
  - D:T
  - D:F
  - D:A
  - D:R
  - D:M
  - D:(W → T)
  - D:(T → F+)
  - D:(A * F+ * R+ → M)
  - D:(M * W → W)

- D:(E:F)
  - D:(E:W)
  - D:(E:T)
  - D:(E:F)
  - D:(E:A)
  - D:(E:R)
  - D:(E:M)
  - D:(E:(W → T))
  - D:(E:(T → F+))
  - D:(E:(A * F+ * R+ → M))
  - D:(E:(M * W → W))
Theory of Research D:F and D:(E:F)

- **D:** (T → F+)
  - **W:** (T → F+)
    - World of processes where features are derived from a theory
  - **T:** (T → F+)
    - Theory of a process of deriving features from a theory
  - **M:** (T → F+)
    - Model of a process of deriving features from a theory
- **(W → T):** (T → F+) = W:(T → F+) → T:(T → F+)
  - A process of creating a theory of deriving features from a world of deriving features from a theory
- **(T → M):** (T → F+) = T:(T → F+) → M:(T → F+)
  - A process of deriving a model of deriving features from a theory from a theory of deriving features from theories
- **(M*W → W):** (T → F+) = M:(T → F+) * W:(T → F+) → W:(T → F+)
  - Injecting a model of deriving features from a theory into the world of deriving features from theories
Evaluating Batory’s F-O Research

- E:(D:F)
  - E:(D:W)
  - E:(D:T)
  - E:(D:F)
  - E:(D:A)
  - E:(D:R)
  - E:(D:M)
  - E:(D:(W → T))
  - E:(D:(T → F+))
  - E:(D:(A * F+ * R+ → M))
  - E:(D:(M * W → W))

- E:(D:(E:F))
  - E:(D:(E:W))
  - E:(D:(E:T))
  - E:(D:(E:F))
  - E:(D:(E:A))
  - E:(D:(E:R))
  - E:(D:(E:M))
  - E:(D:(E:(W → T)))
  - E:(D:(E:(T → F+)))
  - E:(D:(E:(A * F+ * R+ → M)))
  - E:(D:(E:(M * W → W)))
Summary

- Small, simple theories D and E form the basis for laying out a very rich space and an underlying theoretical foundation for SE, SE research, and other design disciplines.
  - Compose D and E into more complex theories to extend and illuminate the space for design disciplines.

- Useful properties:
  - Regularity among the various theories.
  - Levels of abstraction (stratification) within the composed theories providing:
    - Intuitive high level abstractions
    - Explicit low level detailed abstractions

- Used approach to model two very different approaches to theories of software engineering: P and F - and the utility of applying D and E to both of them.