Software Architecture and its relevance for Software Engineering

Dewayne E. Perry
Bell Laboratories
Room 2A-429
600 Mountain Ave
Murray Hill NJ 07974
dep@research.bell-labs.com
www.bell-labs.com/~dep/

Outline

• Software Architecture - Introduction
  – Model of SWA and state of current research
  – Architecture versus design
  – General relevance of architecture
• Relevant Issues for SW Engineering
• Issues of Emerging Significance

Models of SW Architecture

• Perry & Wolf 89/92 model of SWA
• SWA = ( Elements, Form, Rationale )
• Elements: process, data and connecting
• Form is the set of properties of, and relationships among, the elements
• Rationale is the justification for the elements and form

State of Current Work

• Pretty much agree about process, data and connecting elements as first class entities
• Models differ primarily with respect to Form
• Few models pay attention to rationale
• Styles tend to focus on element and form restrictions

Current Approaches to Form

• Configuration
• Type
• Pattern
• Property
Outline

• Software Architecture - Introduction
  – Model of SWA and state of current research
  – Architecture versus design
  – General relevance of architecture
• Relevant Issues for SW Engineering
• Issues of Emerging Significance

Architecture vs Design

• Why separate architecture from design?
• Useful separation of concerns
• Akin to high level design
• Focuses on initial structural issues

Architecture vs Design

• Differences between Architecture and Design
  – Architecture is concerned about higher level issues
  – components vs procedures
  – interactions among components vs interfaces
  – constraints on components and interactions vs algorithms, procedures and types

Architecture vs Design

• Architecture is concerned with a different set of structural issues
  – Large-grained composition vs procedural composition
  – Component interactions (protocols) vs procedural/task interactions (pc, rpc, msgs, etc)
  – Information content vs data types and representations

Outline

• Software Architecture - Introduction
  – Model of SWA and state of current research
  – Architecture versus design
  – General relevance of architecture
• Relevant Issues for SW Engineering
• Issues of Emerging Significance

General Relevance

• Establishes the structure for satisfying system drivers
  – User/Market Requirements
  – Domain requirements
  – Business constraints
  – Product-line constraints
  – Project constraints
General Relevance

• Defines the important structural aspects
  – The load-bearing walls
  – The components, their properties and relationships,
  – The styles of initialization, fault recovery, reliability, etc

General Relevance

• Provides a structural framework for
  – System development,
  – System evolution,
  – Component design and implementation,
  – Asset generation and use/reuse, and
  – System composition

Outline

• Software Architecture - Introduction
• Relevant Issues for SW Engineering
  – Specification
  – Codification
  – Reuse - Product Lines
• Issues of Emerging Significance

Uses of Architectural Specifications

• Prescription vs Description
• Traceability
• Analysis
• Visualization and simulation
• Configuration/Generation

Architectural Prescriptions

• Emphasis on intent, critical aspects
• Tendency towards minimality or incompleteness
• Problem domain emphasis
• Tendency towards high level constraints

Architectural Descriptions

• Emphasis on what exists
• Tendency towards completeness
• Implementation domain emphasis
• Tendency towards detailed descriptions
Traceability

- Rationale is link between architecture and its drivers
  - Non-functionally induced structure
  - Functionally induced structure
- Mapping to design/impl components

Analysis

- Level of analysis depends on
  - the underlying model
  - the expressiveness of the specification language
  - Configuration: standard build
  - Type: compiler technology
  - Pattern: model checking and simulation
  - Property: depends on
    - expressibility
    - decidability

Analysis

- Typical kinds of Analyses
  - Style conformance
  - Consistency and Completeness
    - configuration completeness
    - configuration consistency
    - component - connector consistency (Garlan et al)
- Other functional properties
  - safety properties
  - mismatch detection (Invaradi & Wolf, et al)
  - satisfaction of component by subarchitecture (Moriconi et al)
- Non-functional properties, for example
  - performance
  - reliability

Visualization & Simulation

- Graphical versions of text
- Representation of analyses (Kramer/Magee)
  - Full patterns of interactions
  - Minimization of interactions
- Simulation of event patters (Luckham et al)
- Visualization/simulation of architectural intent
  - Instrumented connectors (Balzer et al)

Configurations

- Build
  - Descriptive specifications
    - configuration model: straightforward
    - other models: need mapping to design/impl
  - Prescriptive specifications
    - determine completeness of arch spec
    - define/generate missing architectural components
    - need mapping to design/implementation
Configurations

- Generate
  - Descriptive specifications
    - Configuration/Type models: not enough information
    - Pattern/Property models: possible to leverage
  - Prescriptive specifications
    - Pattern/Property models useful
    - Need deep understanding of domains for completion
    - Once completed, possible to leverage

Outline

- Software Architecture - Introduction
- Relevant Issues for SW Engineering
  - Specification
    - Codification
  - Reuse - Product Lines
- Issues of Emerging Significance

Codification

- Implementation components
- Type approach
- Patterns approach
- Property approach
- In general, still a long way to go

Codification - Components

- Basic Platforms
  - Common components: GUIs, object mgmt, etc
  - Domain-specific: application-specific platforms
    - first step towards a product line architecture
- Shared Assets
  - Motivation: cost, interval leverage
  - first step towards domain specialization
- Serves as basis for architectural generation

Codification - Type Approach

- Classified existing common components and connectors
- Tendency:
  - Functional classification
  - Solution domain
- Codified styles: restriction of component and connector types
  - For example, pipes and filters

Codification - Pattern Approach

- Design patterns - micro-architectural
- Tends to be informal
- Architectural idioms - closer to type approach
- Styles - defined in terms of patterns
  - event patterns
  - interaction patterns
**Codification - Properties (Batory)**
- Domain-specific architectural assets
  - Components appropriate to the domain
  - Components defined by properties
- Consistent architectural instance created by
  - Component composition on the basis of desired properties
  - Propagating and satisfying the desired properties (ala Perry’s Inscape)

**Codification - Long Way To Go**
- Need non-functional properties
- Understanding of interaction between functional and non-functional properties
- Codification in problem domain
  - Domain-specific templates
  - Applicability of codified solution domain components to problem domain components

**Outline**
- Software Architecture - Introduction
- Relevant Issues for SW Engineering
  - Specification
  - Codification
  - *Reuse - Product Lines*
- Issues of Emerging Significance

**Product Line - Basic Aspects**
- Begin with product instances
  - legacy based
  - use architecture recovery processes
- Focus on appropriate business domain
  - use domain specific architectural processes
  - map from recovered to domain architecture
- Abstract/Generalize to Product Line Architecture

**Product Line - Overview**

**Product Line - Issues**
- Product Line Reference Architecture
- Product Line Processes
- Asset Base
- Supporting Technology
- Organizational Issues
Product Line - Ref Architecture
- Domain-specific prescription or description
- Parameterized architectural components
- Refinement into sub-architectures
- Style descriptions for
  - critical architectural aspects
  - orthogonal aspects - eg, initialization, fault recovery, etc

Product Line Processes
- Create/evolve the reference architecture
- Create/evolve architectural instances
  - instantiate and provision
  - configure and generate
- Create/evolve asset base
  - shared components
  - specialized components
- Use asset base for architectural instance/impl

Asset Base
- Design component descriptions
  - common interfaces
  - common implementations
  - product-specific implementations
- Various supporting platforms
- Product specific components

Supporting Technology
- Architecture
  - Analysis - sufficiency, satisfaction
  - Instantiating, provisioning, customization
  - Generation/configuration
- Design/Implementation
  - Architecture satisfaction analysis
  - Component composition/analysis
  - Connector optimization
  - Run-time generation

Organizational Considerations
- Architecture/Asset base
  - across product lines
  - product line specific
  - product specific
- Supporting technology
  - global to the company
- Processes - support multiple product lines

Outline
- Software Architecture - Introduction
- Relevant Issues for SW Engineering
- Issues of Emerging Significance
  - Styles
  - Connectors
  - Dynamics
Styles
• An incomplete architectural prescription
• Focuses on certain aspects of the architecture
  – architectural elements
  – formal characteristics
  – constraints on architectural elements
  – constraints on formal characteristics

Styles
• Problem: Restrict the architectural structure
  – for example, strict layering of the architecture
• Solution: layered architecture style
  – constrain the interactions
    • any interaction at elements on the same level
    • no interactions at more than one level away
    • level below: initiate interactions only
    • level above: react interactions only

Styles
• Problem: multi-dimensional organization
  – Select one as primary, others as secondary
• Solution: Styles for the secondary dimensions
  – primary dimension: architectural elements
  – secondary dimensions then distributed over primary
  – styles define the characteristics of the distributed dimensions

Styles
• Useful rule of thumb: a style for a domain
• Problem: multiple domains in any significant architecture
• Challenge: integrating the styles consistently

Outline
• Software Architecture - Introduction
• Relevant Issues for SW Engineering
• Issues of Emerging Significance
  – Styles
  – Connectors
  – Dynamics

Connectors
• Primarily thought of means of communication
  – procedure call, remote procedure call
  – message passing with various levels of service
  – constraints on structure and directions - pipes
  – constraints on quality of service - persistence
Connectors

• Extremely useful in this context
  – separates computation from interaction
  – can change some non-functional characteristics by changing connectors
    • from prototype to embedded system via connectors (Tracz)
    • improve performance via connector optimization

Connectors

• Can be used as means of mediation
  – govern access to shared data structures
  – provide synchronization, exclusion
    • critical sections
    • monitors
  – determine what is allowed and when
    • readers/writers policies
    • path expressions

Connectors

• Extremely useful in this context
  – separates mediation control from computation
    • localizes synchronization and exclusion control
    • localizes operational policies
  – separate mediation from communication
  – compose communication and mediation connectors

Connectors

• Can be used as means of coordination
  – determine control of computation
    • elements of control in communication
    • elements of control in mediation
  – control loci of execution
  – control delivery of data

Connectors

• Extremely useful in this context
  – separate aspects of control from computation
  – instrumented connectors (Balzer)
    • mutual invocation - like coroutines
    • coordination of computation results and data delivery
  – fault tolerance
    • separate exception handling as a plane of control
    • becomes compositional not integral

Outline

• Software Architecture - Introduction
• Relevant Issues for SW Engineering
• Issues of Emerging Significance
  – Styles
  – Connectors
  – Dynamics
Dynamics

- Allowed dynamic changes
  - creation/destruction of components and connectors (Kramer & Magee)
  - to respond to dynamic system requirements
- Appropriate support for
  - distribution independence
  - dynamic linking, registration (Taylor et al)

Conclusions

- Separates out useful level of concern
  - problem domain meets implementation domain
- Defines important constraints on the system
- Basic structure of the system
- Means of capitalizing on assets
- Moves us from integral to compositional
- Integrates composition with generation