Sustainable Architecture

A Short Introduction

Alar Raabe

Teemad

- Kas jätkusuutlik arhitektuur on võimalik
- Milline on vähim võimalik reeglite kogu arhitektuuri jätkusuutlikuse tagamiseks ja mida see peaks reguleerima
- Kuidas tagada piisavalt kõrge ärijuhtkonna huvi (ja/või asjatundlikus) ettevõtte arhitektuuri vastu (huvi otsuste pikaajaliste mõjude vastu)
- Kuidas ettevõttes toime tulla konsultantidega

Issues

- Is the sustainable architecture possible?
- Which is the smallest possible set of rules/principles that assure the sustainability of architecture and what is should regulate?
- How to make (sufficiently high-level) management of the enterprise interested (knowledgeable) of enterprise architecture (interested of long-term effects of their decisions)?
- How to handle consultants in the enterprise?

Sustainability

sustainability (lat. sustinere) – the capacity to endure (to hold up)

- **Let's look the opposite case – what makes an architecture (a system) not sustainable**
  - large "technical debt" that is such change (corruption) of the original architecture, that non-functional qualities of the system are significantly reduced (for example modifiability)
  - loss of architectural knowledge, causing the maintenance of the system to become difficult or impossible
  - change of the external environment in such extent that the architecture doesn't allow the system to be adapted to this change

- **Growth of the entropy (measure of disorder) in the systems is unavoidable if tere is not a constant effort applied to reduce it**
  - that leads us to the requirement that in the sustainable architecture we must have mechanisms to counteract the growth of entropy or even reduce it
  - sustainable architecture keeps the entropy (measure of disorder) of the system constant or reduces it

Emerged Architecture – Big Ball of Mud !

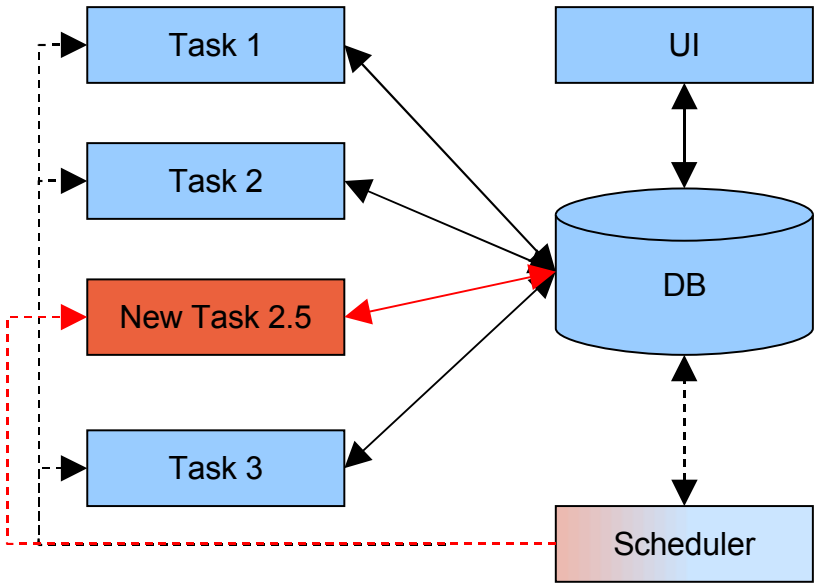
Complexity increases rapidly until it reaches a level of complexity just beyond that with which we can comfortably cope

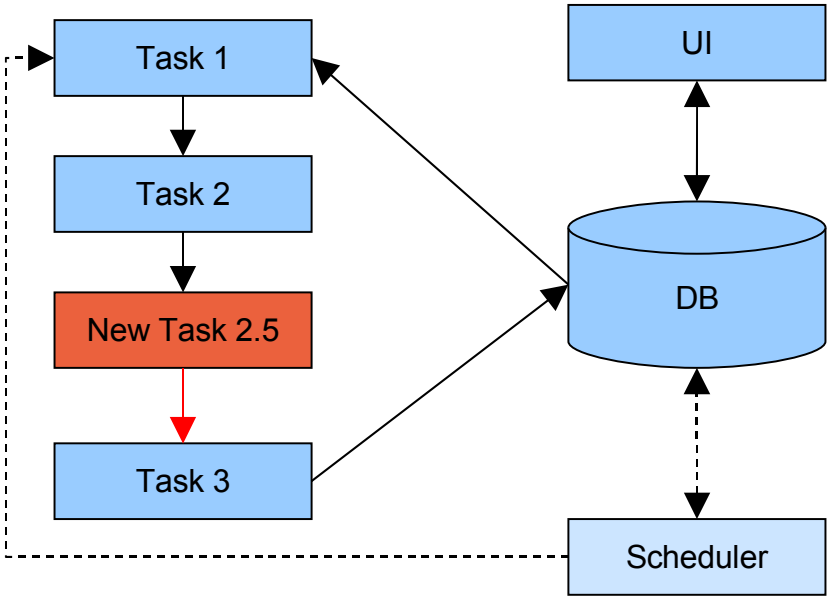
Cunningham

- Emerges from
  - Throwaway code, Piecemeal growth, Keep-it-Working,
  - Shearing layers, Sweeping it under the rug
- Forces corresponding to emergence
  - Time – designing architecture takes time
  - Cost – designed architecture costs and is long-time investment
  - Experience and skill – designing architecture requires know-how
  - Complexity and scale of the problems
  - Change – predicting future change requires vision and courage
  - Organization – architecture reflects organization (Conway's law)
- Advantages – ***mostly business concerns !***
  - Quick to make → Time-to-Market
  - Cheap to make → Cost vs. Benefit
  - Does not need governance – just emerges
  - Does not need skills
- Disadvantages – ***mostly IT concerns !***
  - Maintainability – difficult and costly to maintain
  - Modifiability – hard and dangerous to change
  - Testability – difficult to test

Different Architectures – Different Properties

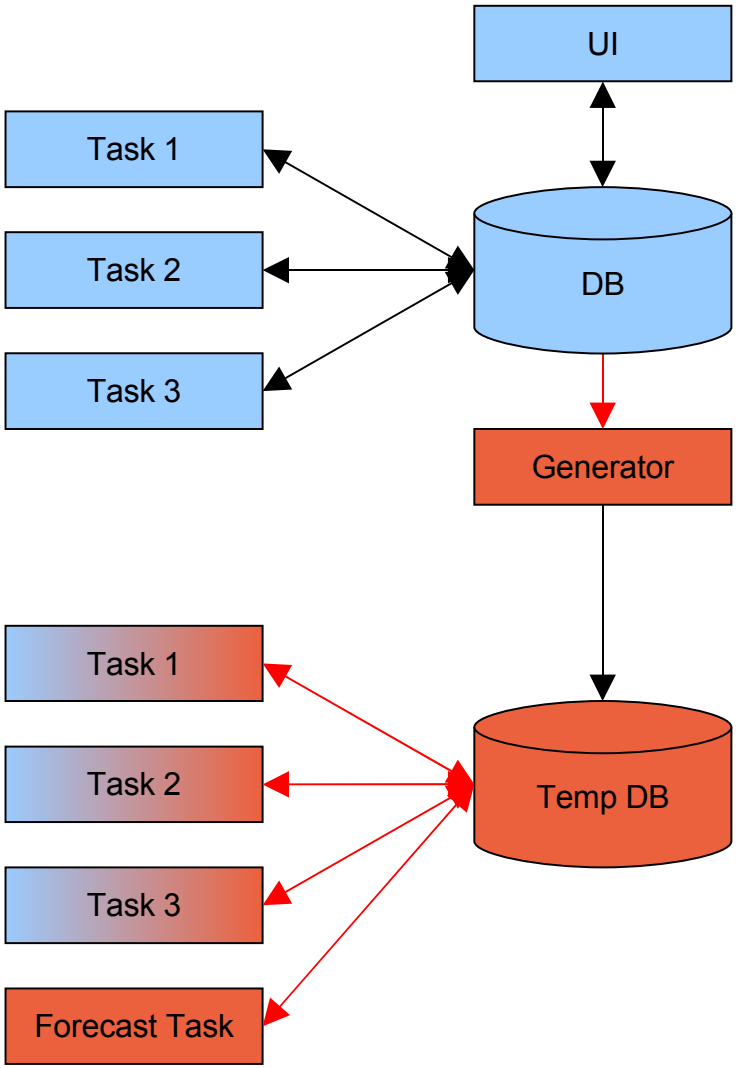
adding new task

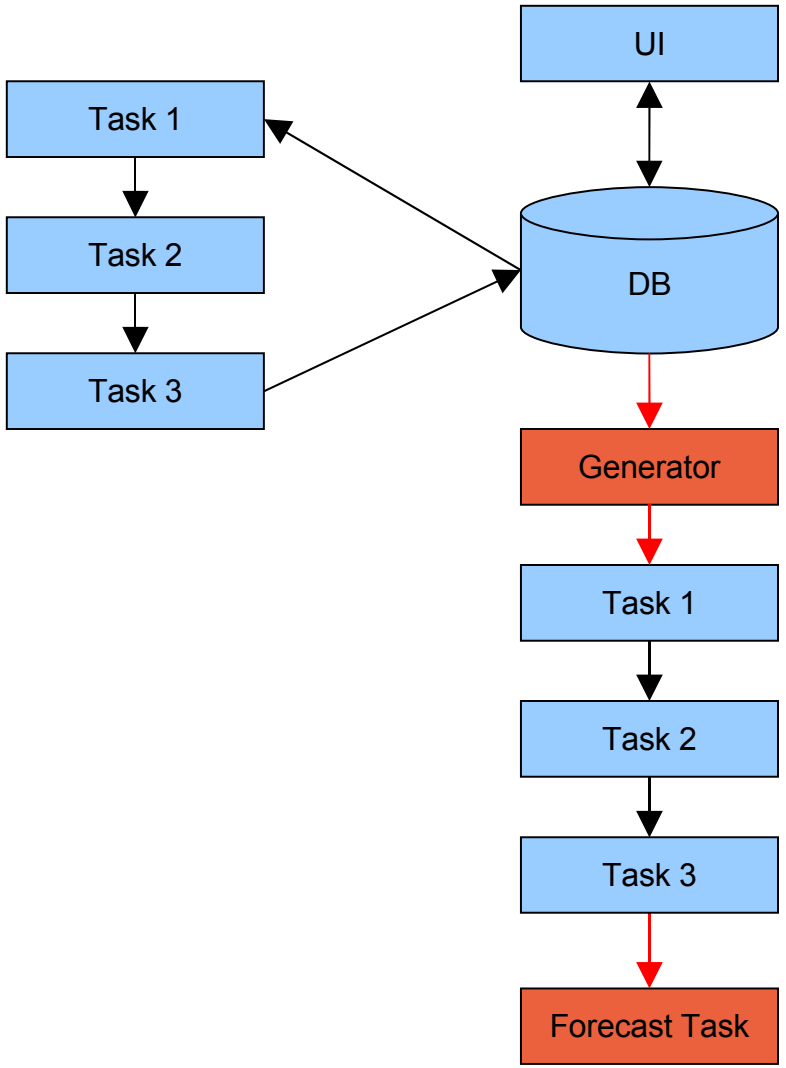




Different Architectures – Different Properties

adding forecasts of portfolio





Controlling the Architecture

> Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations
>
> Conway (1968)

- **Use Conway's law**
  - enterprise organization should be designed such that we would like the enterpise IT architecture to become
  - there must always exist an interested party for developing certain architecture feature/element (e.g. developing the architecture in certain way must be in somebody's interest)
- **Limit the resources in suitable manner would stimulate thinking and (as a consequence) reduction of complexity, increase of reuse, and simplification of maintenance**
  - that is architecture can be developed by the owner/steward of necessary resources
  - in case we have a governance body for architecture like architecture committee, it must own or command resources
- **Ownership of main (architectural) principles by top management (they must belong to the enterprise identity and value system)**
  - only then they will influence the architectural decisions;

Controlling the Architecture

- **Differentiat actions according to the governance model**
  - in the enterprises with strong central governance (power) it is possible to design the architecture accordin to certain goals, and the result would be foreseeable
  - but in the enterprises with weak central governance (power) or completely decentralized enterprises it would only be possible to create favorable situation for architecture to emerge and develop (grow), and the result would not be foreseeable nor guaranteed

- **Differentiate capital investments**
  - the part of the architecture, which function is most static (i.e. infrastructure/platform) could afford the biggest capital investments – to stay long time without changes it must perfrom very generic functions (as an opposite to the frequently changing parts of the architecture which could perfrom very specific functions)
  - the more variable/volatile is the function, the smaller should be the capital investments

- **Limit the amount of "technical debt" allowed for the IT architecture**
  - large technical debt limits the options of change and "takes over" control of the architecture.

Sustainability Evaluation of Software Architectures by Heiko Kozioleko (ABB)

- A software-intensive system is long-living if it must be operated for more than 15 years
- A long-living software system is sustainable if it can be cost-eciently maintained and evolved over its entire life-cycle
  - the opposite of a sustainable software system is a longliving system that cannot be adapted to changing requirements and environments due to unjustiable costs or even technical infeasibility
  - the architecture of a sustainable system may evolve during its life-cycle, but the fulllment of customer requirements within timing, budget, and quality constraints must be assured
- Sustainability
  - comprises the attributes maintainability (i.e., analysability, stability, testability, understandability), modiability, portability, and evolvability
  - can be achieved through adherence to design principles (e.g., modularity, separation of concerns, conceptual integrity) throughout the entire lifecycle

(Lean) Enterprise Principles by Deborah Nightingale (MIT)

- Adopt a Holistic Approach to Enterprise Transformation
- Identify Relevant Stakeholders and Determine their Value Propositions
- Focus on Enterprise Effectiveness before Efficiency
- Address Internal and External Enterprise Interdependencies
- Ensure Stability and Flow within and across the Enterprise
- Cultivate Leadership to Support and Drive Enterprise Behaviors
- Emphasize Organizational Learning

Entropy

- 3 Interpretations
  - Irreversibility: engines produce unrecoverable heat
    - The Arrow of Time: Closed systems entropy always increases as the Universe's
  - Measure of the disorder: Kid's room, engineer desk…
  - Measure of ignorance: We are part of the system: Disorder prevents understanding
- Entropy of an open system can increase or decrease
  - At the expense of the surrounding system
- Information Entropy (Shannon)
  - The minimum length of a message for a given meaning
  - Affected by coding, noise, redundancy
  - Could be generalized to measure the "effectiveness" of information processing

Ten Signs of Enterprise Entropy

1. Most funded projects are fun to build, but do not directly support key business drivers
2. The quality improvement process has become so internalized that a high percentage of funded projects are creating very high-quality redundant functions, data stores and interfaces
3. No one has noticed the linkage between the measurements used to indicate the overall health and success of the organization — shareholder value, high quality/low error rates, customer satisfaction — with the 22 inconsistent, overlapping customer data stores and the high level of customer complaints about receiving duplicate mailings
4. To support "Buy Vs. Build," each Line of Business has purchased its own trouble-reporting system — and server to host it
5. There is a governance process, but basically, any tall person with a loud voice can build a new customer data store
6. There are at least several effective, well-managed work intake processes, with highly trained project managers each tracking their own overlapping, competing projects
7. There is a formal Systems Development Methodology — somewhere…
8. The IT organization structure looks like a bad module design
9. When projects are late/over budget/irrelevant, there is usually stunned surprise (How could this have happened?)
10. The corporate data model just celebrated year ten of its development, but the only cake-eaters were the corporate data modelers…

Agile Enterprise

- **Fred Brooks:**
  - All repairs tend to destroy the structure, to increase the entropy and disorder of the system. Less and less effort is spent on fixing the original design flaws: more and more is spent on fixing flaws introduced by earlier fixes.

- **John Zachman:**
  - I have a feeling that the reason the whole enterprise has to embrace enterprise architecture is because of entropy: the energy that enterprises have to spend just to keep the system working rather than spending it on doing productive work. This is the second law of thermodynamics: entropy increases over time.
  - When the structure of the model supports the intent, when everything supports the layer above it, then you get clarity and flexibility. You want real flexibility? You separate the independent variables. You don't get flexibility by increasing the granularity. That does not make it flexible. Process must be separate from inventory. Logic must be separate from technology. ... When you can change one thing without touching the others, you have flexibility.

Thank You!

Architecture

Architecture is about:

- Durability (*firmitas*)
- Utility (*utilitas*)
- Beauty (*venustas*)

Vitruvius
(Rome, 1 BC)

- Merriam-Webster :: Architecture (n)
  - art or science of building
  - unifying or coherent form or structure
  - manner in which the components of the system are organized and integrated

- Wikipedia :: Architecture
  (Greek: *αρχιτεκτονική* and Latin: *architectura*)
  - 2011
    - art and science of designing (*buildings and other physical*) structures
    - style and method of design and construction of (*buildings and other physical*) structures
    - …
  - *2009*
    - *…*
    - *as documentation, usually based on drawings, architecture defines the structure and/or behavior of a system that is to be or has been constructed*

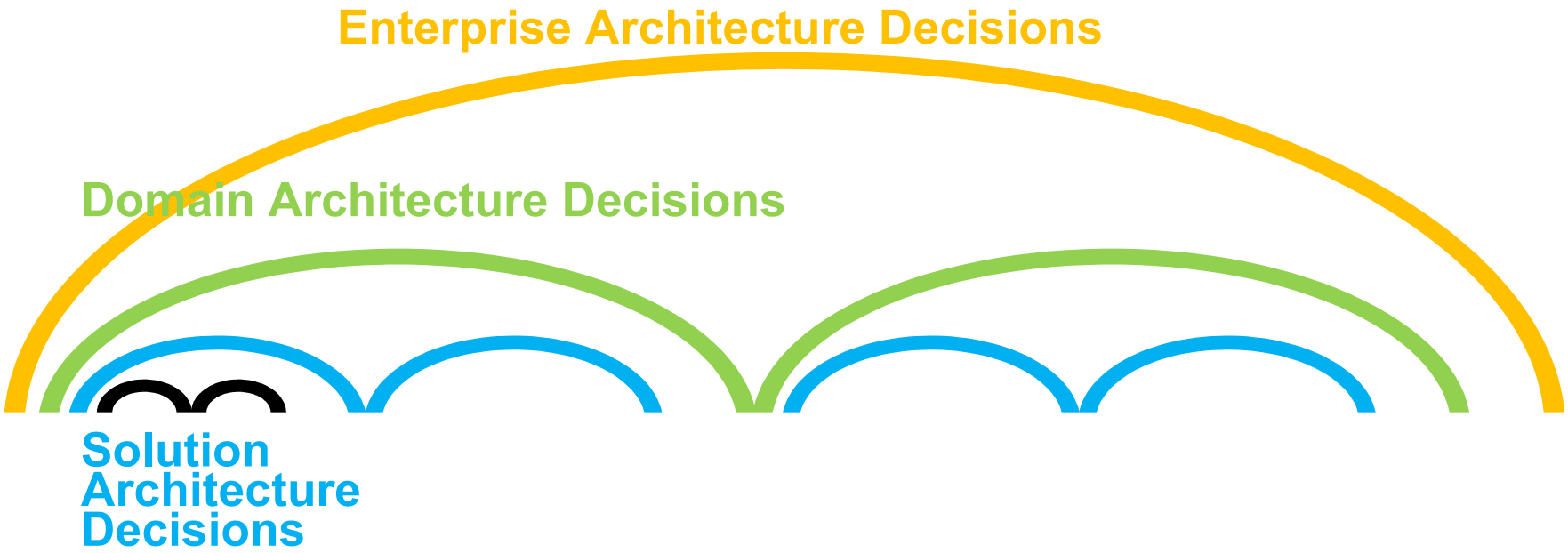
What is System Architecture

architecture is a model of system and architecture description is a model of architecture

- System Architecture is a
  - *fundamental conception* of a system in its
  - ***environment*** embodied in
  - ***elements***, their
  - ***relationships*** to each other and to the environment, and
  - ***principles*** guiding software system design and evolution
- System Architecture Description is a
  - collection of ***related (corresponding) models***, organized into cohesive groups of
  - *synthetic* (constructed) or *projective* (derived) ***views***, defined by ***viewpoints*** according to the related set of ***concerns*** *(in architecture framework)*
- System Architecture Model is
  - ***work product*** that can be used to answer questions about the system
    - M. Minsky 1968: “to an observer B, an object A* is a model of an object A to the extent that B can use A* to answer questions that interest him about A”
    - IEEE SE VOCAB: an interpretation of a theory for which all the axioms of the theory are true, or a semantically closed abstraction of a system or a complete description of a system from a particular perspective

Different Architecture Levels – Decision Scopes

- Enterprise Architecture – a **holistic view** on whole enterprise
  - a description of the enterprise that provides a common understanding and a **formal link between strategic objectives and tactical execution**



Measuring Value of Software Architecture

Focus on quality and cost will decrease
Focus on costs and quality will decrease

W. E. Deming

- Value of Software Architecture
  - Cost of realization of risks compared to cost of architecture

$$value_{arch} = \sum_{i=1}^{n} \left(cost_{risk}\left(concern_i \right) \right) - cost_{arch}$$

- Value of Software Architecture Description
  - Cost of performing activities without architecture description compared to cost of documenting architecture

$$value_{arch.desc} = \sum_{i=1}^{n} \left(cost_{performing}\left(activity_i \right) \right) - cost_{arch.desc}$$

Real Options for Valuation of Software Architecture

- Option is
  - A right, but not obligation to make a decision in the future
  - Difference from financial option (American call) – might be exercised multiple times
- Applicable when there is
  - ***Uncertainty***
  - ***Business goal***
    - Uncertainty is important for managing or achieving a business goal
  - ***New information***
    - New information should be exploited when it comes available
  - ***Action*** today should create
    - Possibility of ***future design choices***
    - Possibility of ***future value***
- Strategic Value with Real Options

$$NPV_{strategic} = NPV_{traditional} + Value_{real.options}$$

- Valuation of real options
  - Decision trees with probabilities (Markov processes)
  - Dynamic programming algorithms
  - Monte Carlo simulations

Economic Value of Architecture Decisions

- Real Option Theory (Sullivan)
  - Qualitative Design Principles
    - If at any time, the NPV of future profits is at least by value of investment more than the direct costs, then commit to the design decision, otherwise not
    - If the expected PV of the future profits that would flow from choice exceeds the direct cost of implementing it, then implement the choice, otherwise implement some other choice
    - If the expected PV of future profits that would flow from restructuring exceeds the direct cost of restructuring, then restructure, otherwise do not
    - If the cost to effect a software decision is sufficiently low, then the benefit of investing to effect it immediately outweighs the benefit of waiting, so the decision should be made immediately
    - With other factors, including the static NPV, remaining the same, the incentive to wait for better information before effecting a design decision increases with the risk (i.e. with the spread in possible benefits)
    - The incentive to wait before investing increases with the likelihood of unfavorable future events occurring
    - All else being equal, the value of the option to delay increases with variance in future costs

Terms (Glossary)

ISO/IEC 42010

Term	Definition
architecture	fundamental conception of a system in its environment embodied in elements, their relationships to each other and to the environment, and principles guiding system design and evolution
architecture decision	choice made from among possible options that addresses one or more architecture-related concerns
architecture description	collection of work products used to describe an architecture
architecture model	work product from which architecture views are composed
architecture rationale	explanation or justification for an architecture decision
architecture view	work product representing a system from the perspective of architecture-related concerns
architecture viewpoint	work product establishing the conventions for the construction, interpretation and use of architecture views
architecture-related concern	area of interest in a system pertaining to developmental, technological, business, operational, organizational, political, regulatory, social, or other influences important to one or more of its stakeholders
environment	context determining the setting and circumstances of developmental, technological, business, operational, organizational, political, regulatory, social and any other influences upon a system
model correspondence	relation on two or more architecture models
stakeholder	individual, team, organization, or class thereof, having concerns with respect to a system
purpose	*{one of system concerns}*
system	*{a conceptual entity defined by its boundaries}*

Architectural Design Decisions

- Kinds of Architectural Design Decisions
  - Existence Decisions (*ontocrises*)
    - Structural decisions
    - Behavioral decisions
    - Ban or non-existence decisions (*anticrises*)
  - Property Decisions (*diacrises*)
    - Constraints
    - Design rules
    - Guidelines
  - Executive Decisions (*pericrises*)
    - Organizational decisions
    - Process decisions
    - Technology decisions
    - Tool decisions
- Attributes of Architectural Design Decisions
  - Epitome (the Decision itself)
  - Rationale ("why")
  - Scope
  - State (idea, rejected, tentative/challenged, decided, approved)
  - Author, Time-Stamp, History
  - Categories (usability, security, …)
  - Cost
  - Risk
- Relationships between Architectural Design Decisions
  - Constraints
  - Forbids (Excludes)
  - Enables
  - Subsumes
  - Conflicts with (mutually excluding)
  - Overrides
  - Comprises (is made of, decomposes into)
  - Is bound to (strong)
  - Is an alternative to
  - Is related to (weak)
  - Dependencies
- Relationship with External Artifacts
  - Traces to
  - Does not comply with