Software Architecture

Theory and Practice

Evolveum

Radovan Semančík

April 2018

Who Am I?

Ing. Radovan Semančík, PhD.

- Software Architect at Evolveum
- Architect of Evolveum midPoint
- Apache Foundation committer
- Contributor to ConnId and Apache Directory API

Poll

Who wants to be:

1. Coder/developer
2. Software designer/architect
3. Manager

So, you wanna be an architect?

What Does Software Architect Do?
Theory

- Draw diagrams (UML anyone?)
- Design great and important systems
- Be a big boss

What Does Software Architect Do?
Practice

- ~~Draw diagrams (UML anyone?)~~

  … implement it too. And test. And document.

- ~~Design great and important systems~~

  … more like databases and JavaScript.

- ~~Be a big boss~~

  … in fact do many things by yourself.

architecture

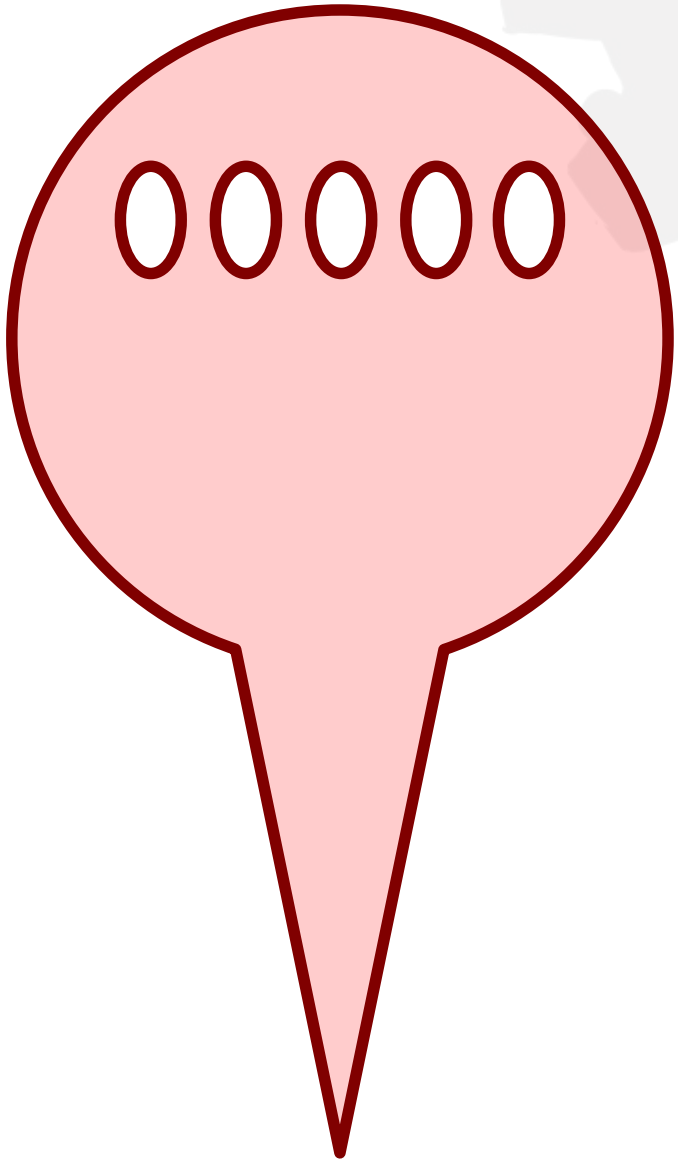
The art or science of building; especially, the art of building houses, churches, bridges, and other structures, for the purposes of civil life; -- often called civil architecture.

Construction, in a more general sense; frame or structure; workmanship.

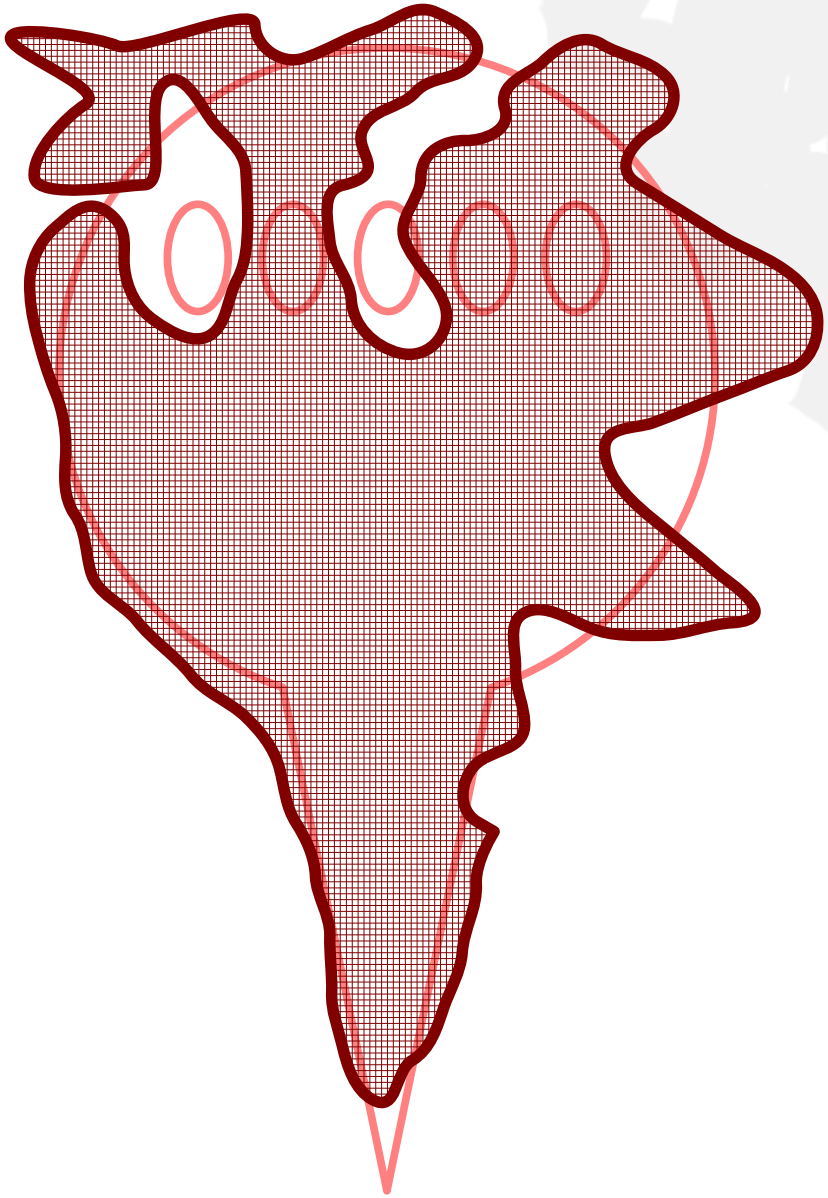
Webster, 1913

How it all works in practice ...

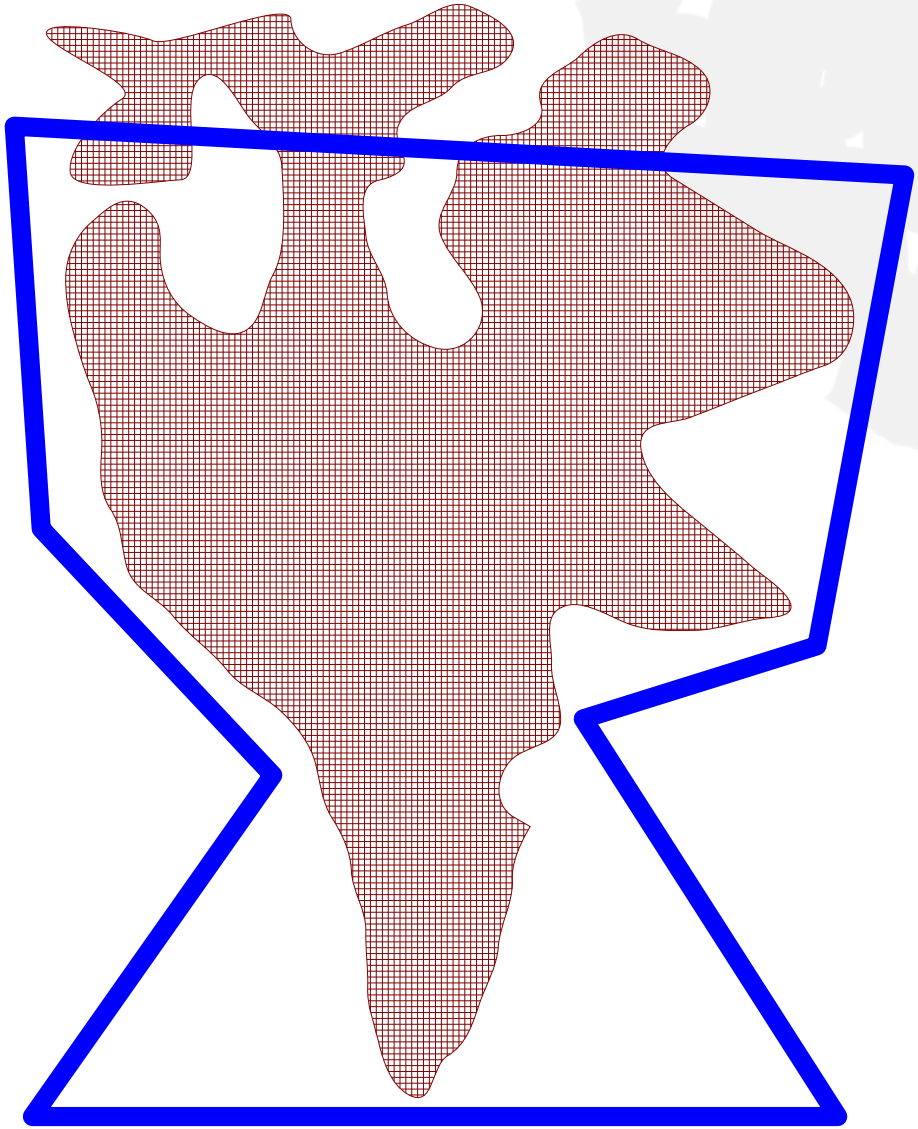
What Client Wanted



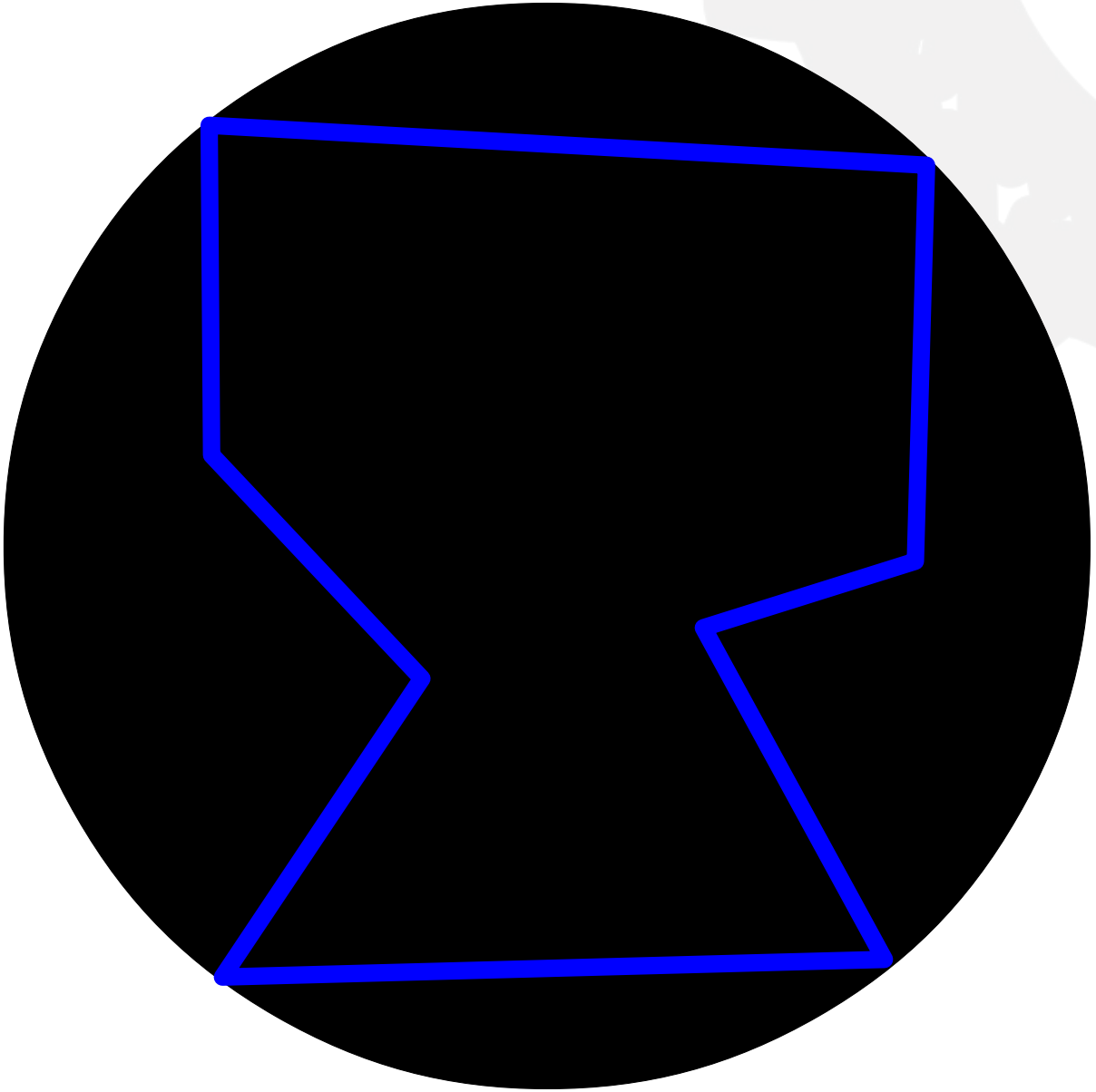
What Client Described



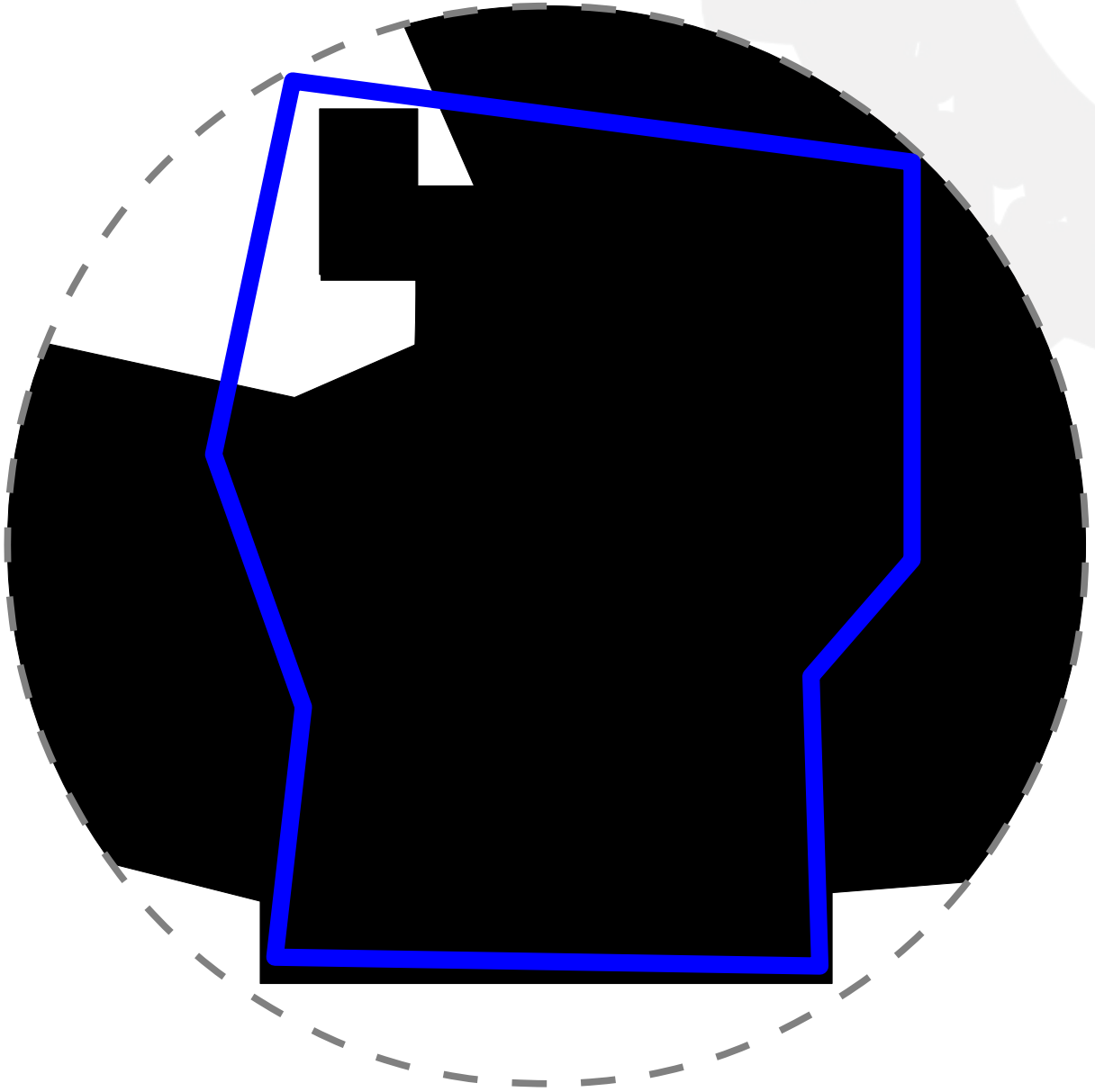
How Architect Understood



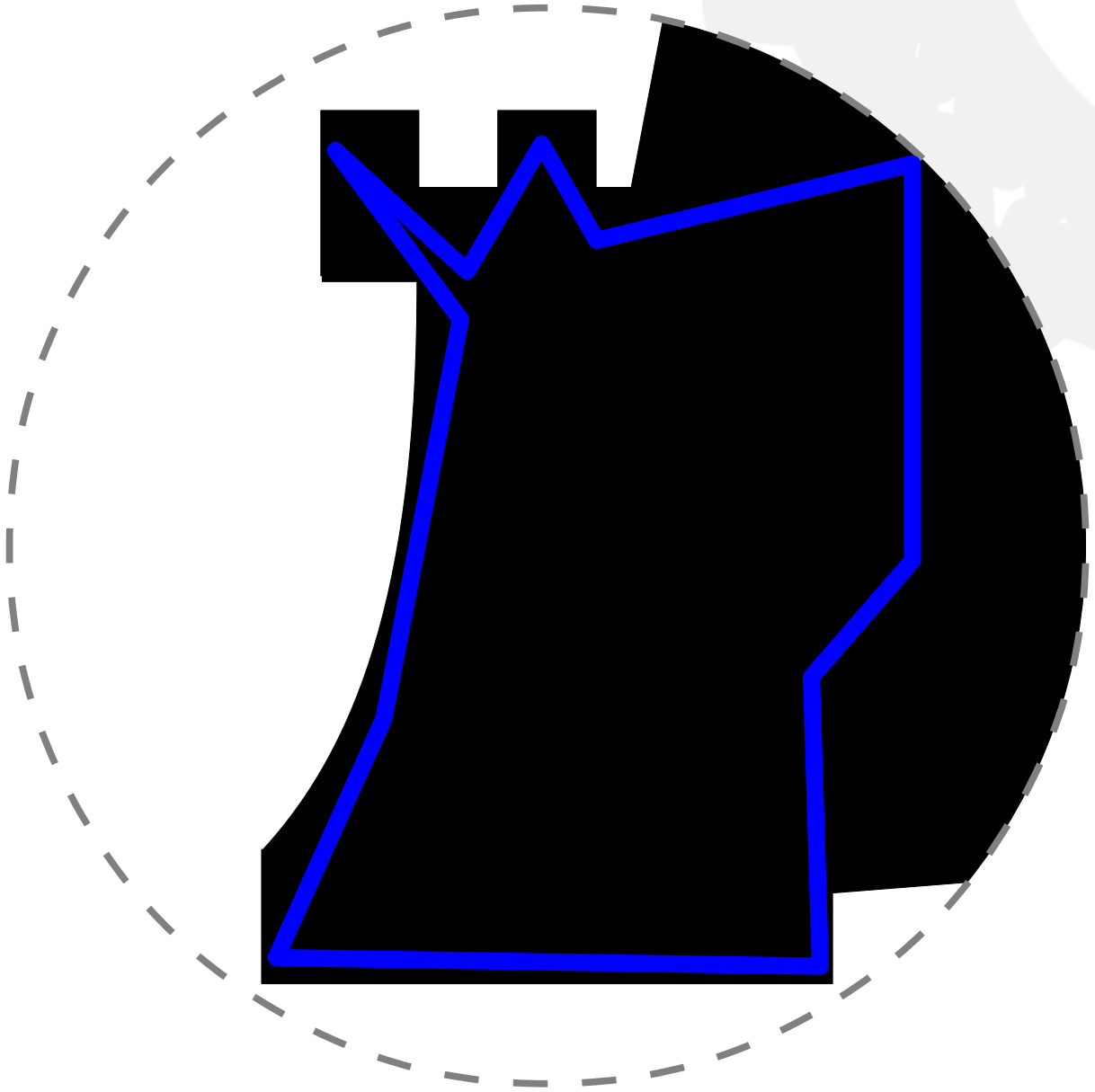
Empty Set of Constraints



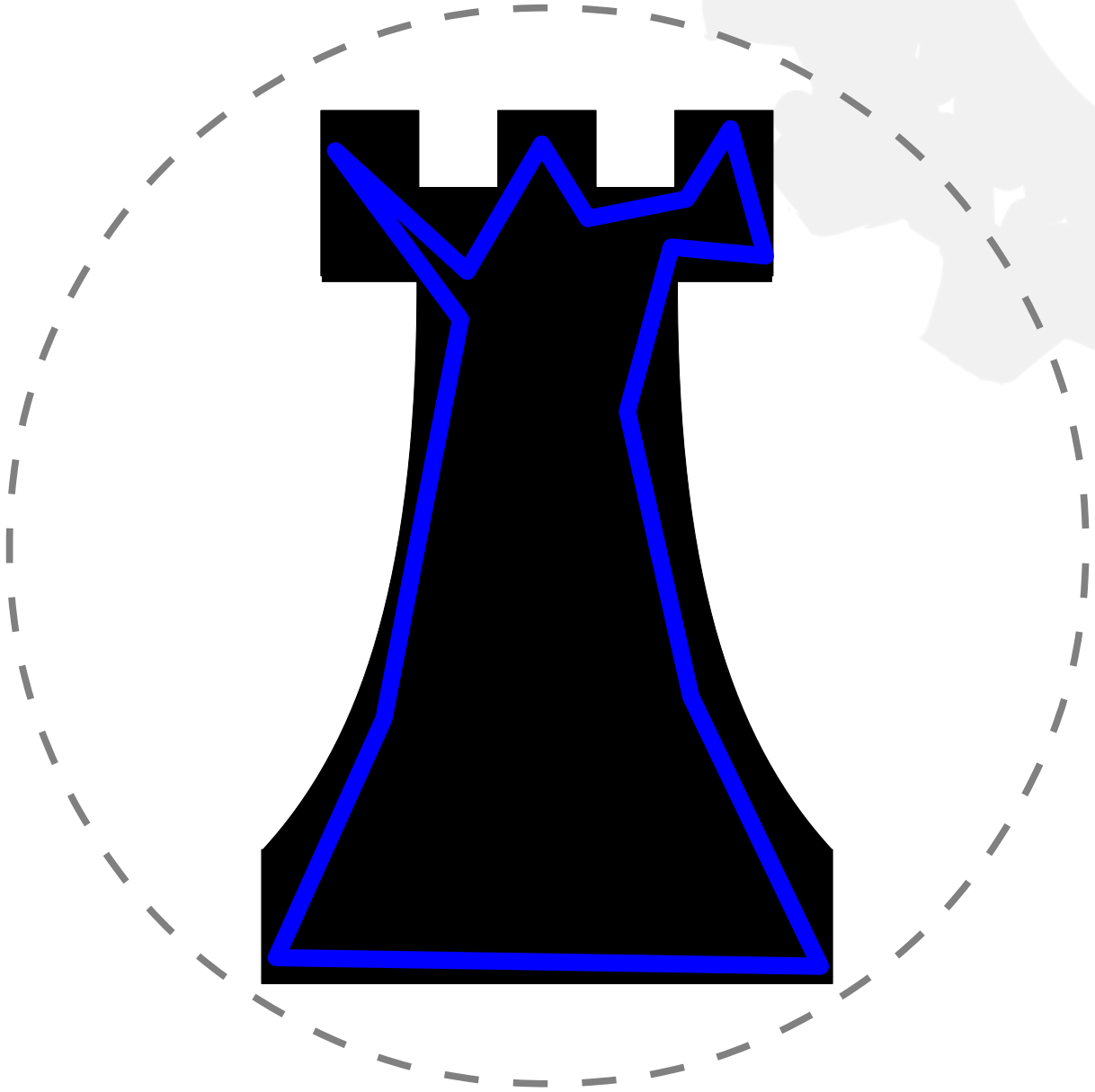
Adding Constraints



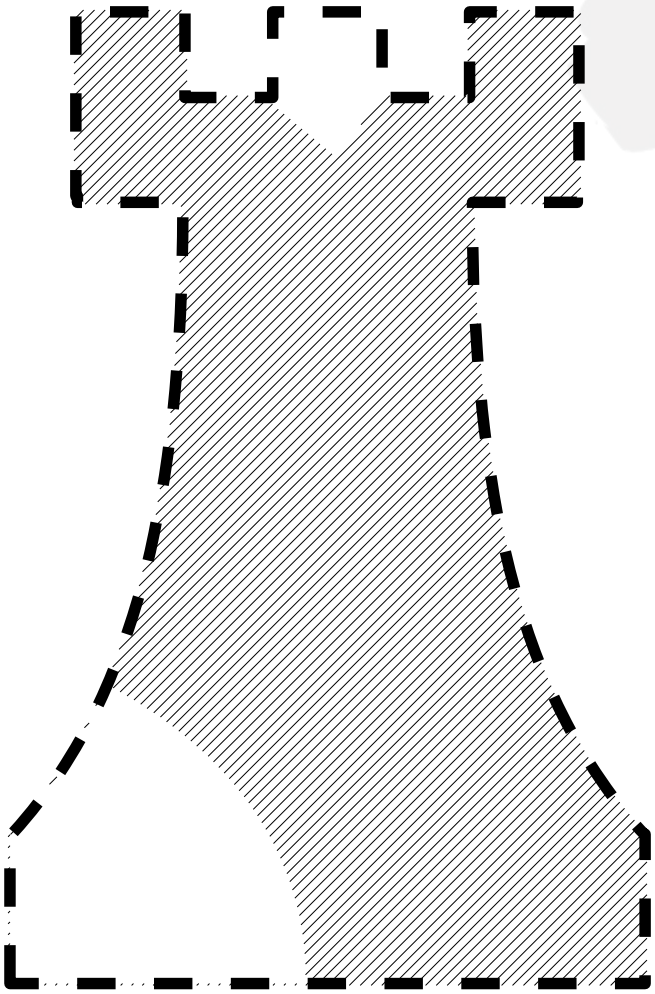
Adding Constraints



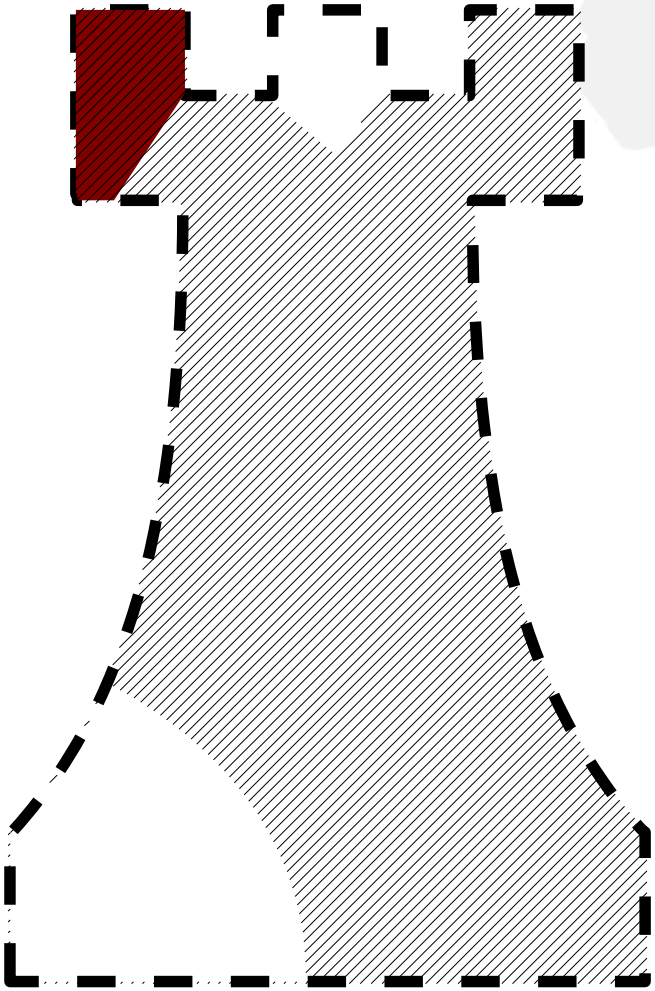
Architecture Finished



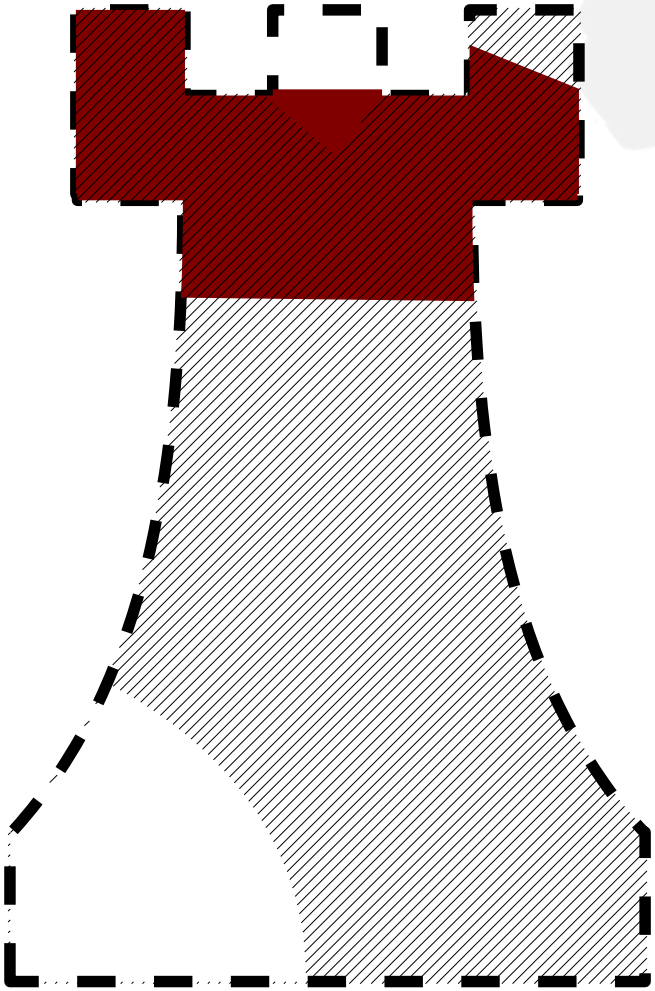
Architecture Documented



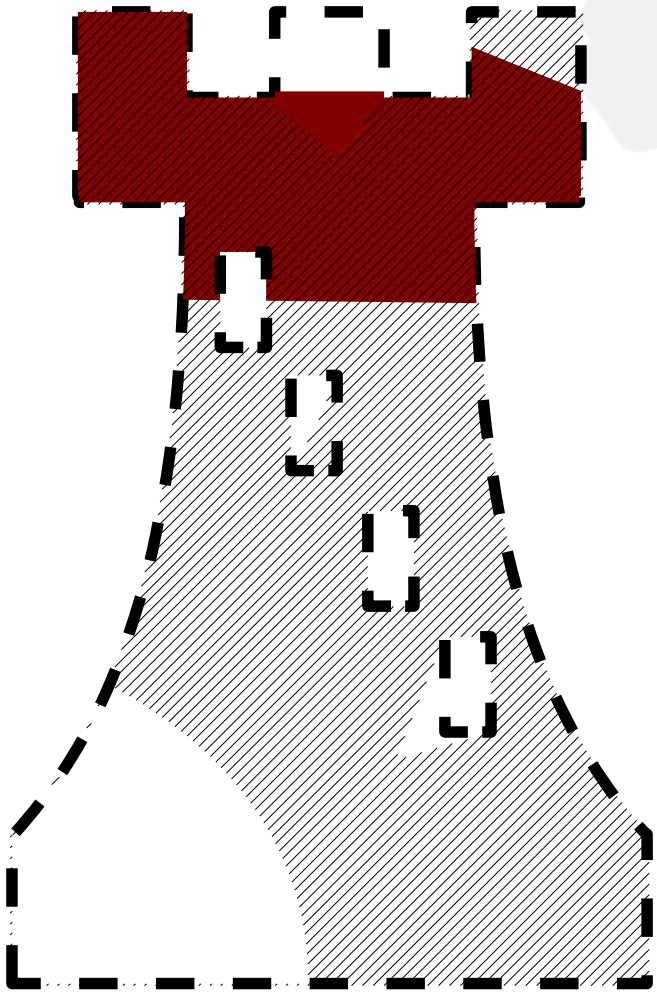
Start of Development



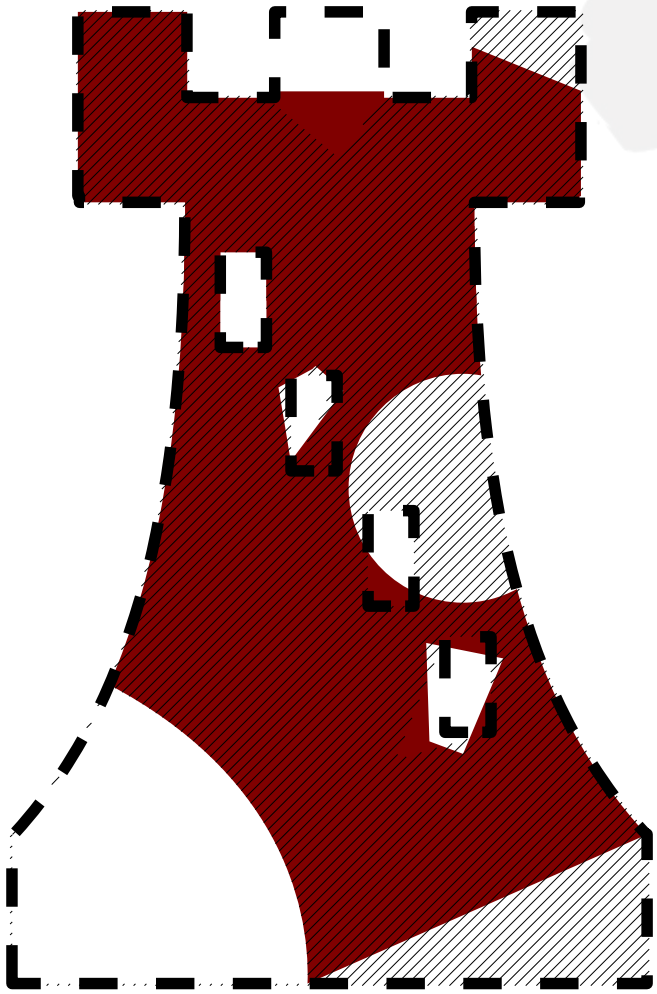
Development



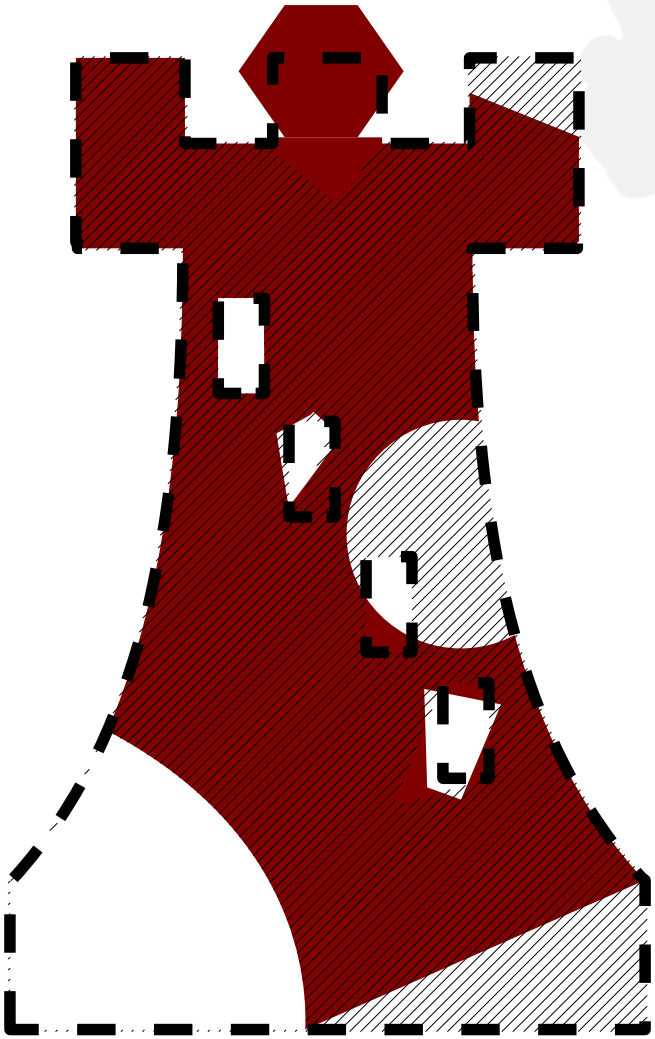
Architectural Issue Discovered



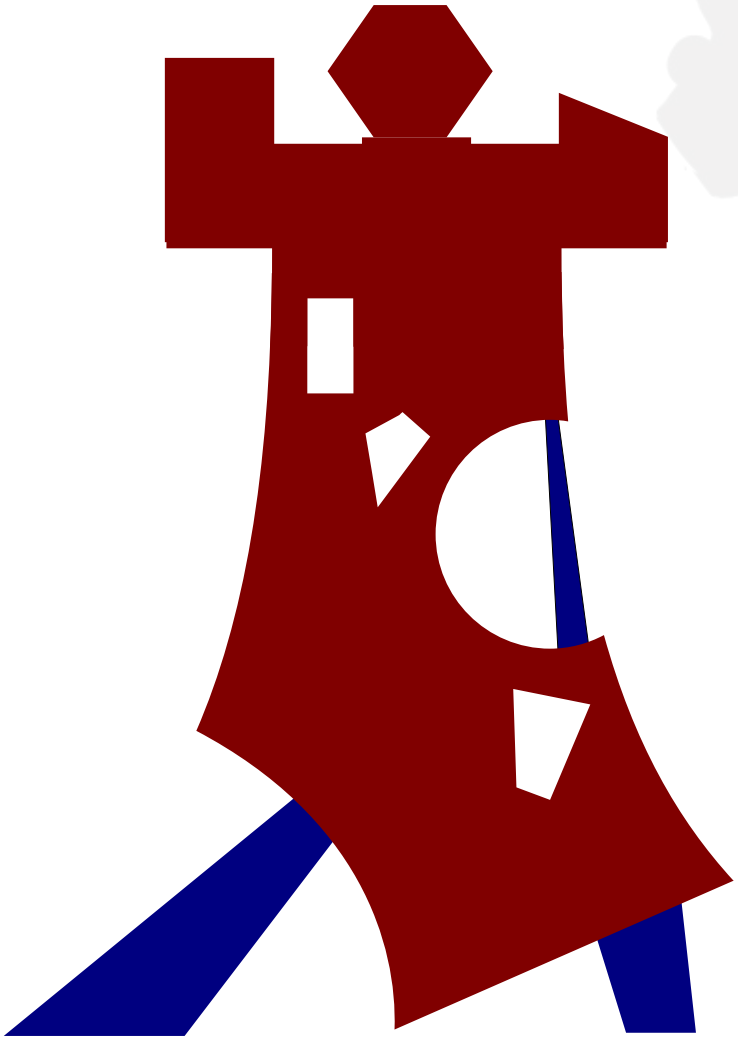
Development



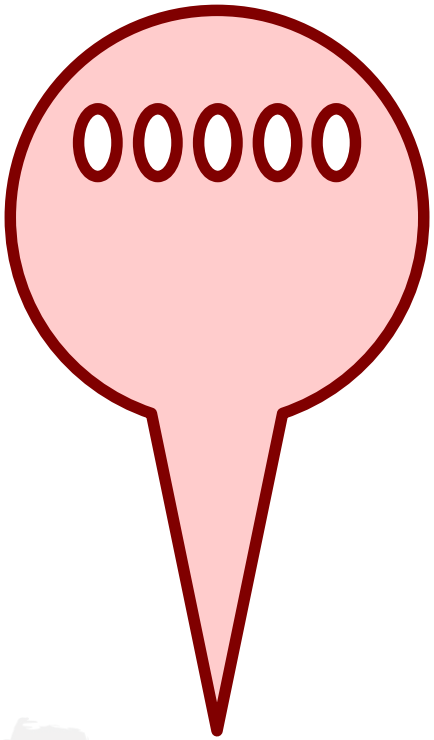
Development Finished



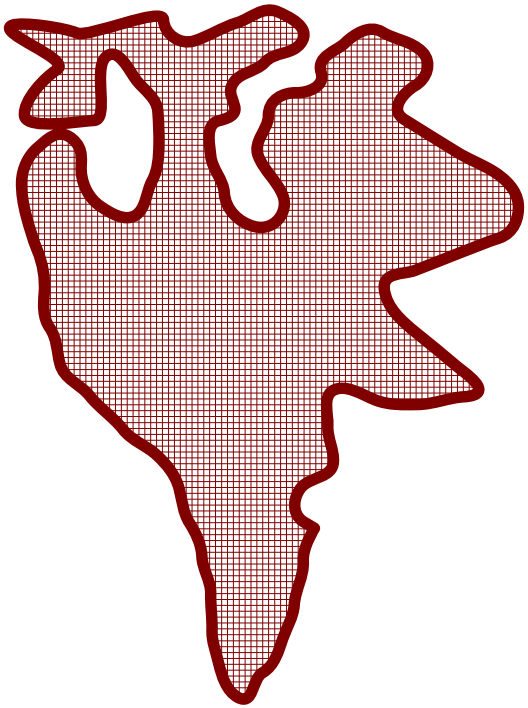
Delivery



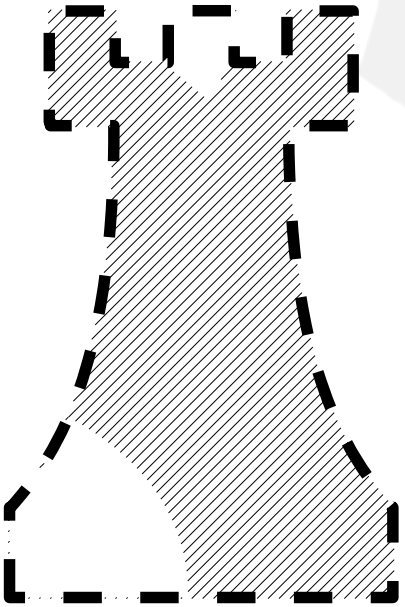
Morphing the System



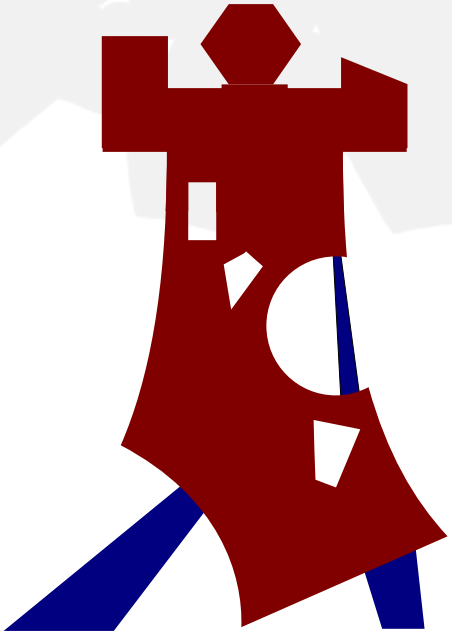
Desired



Described



Designed

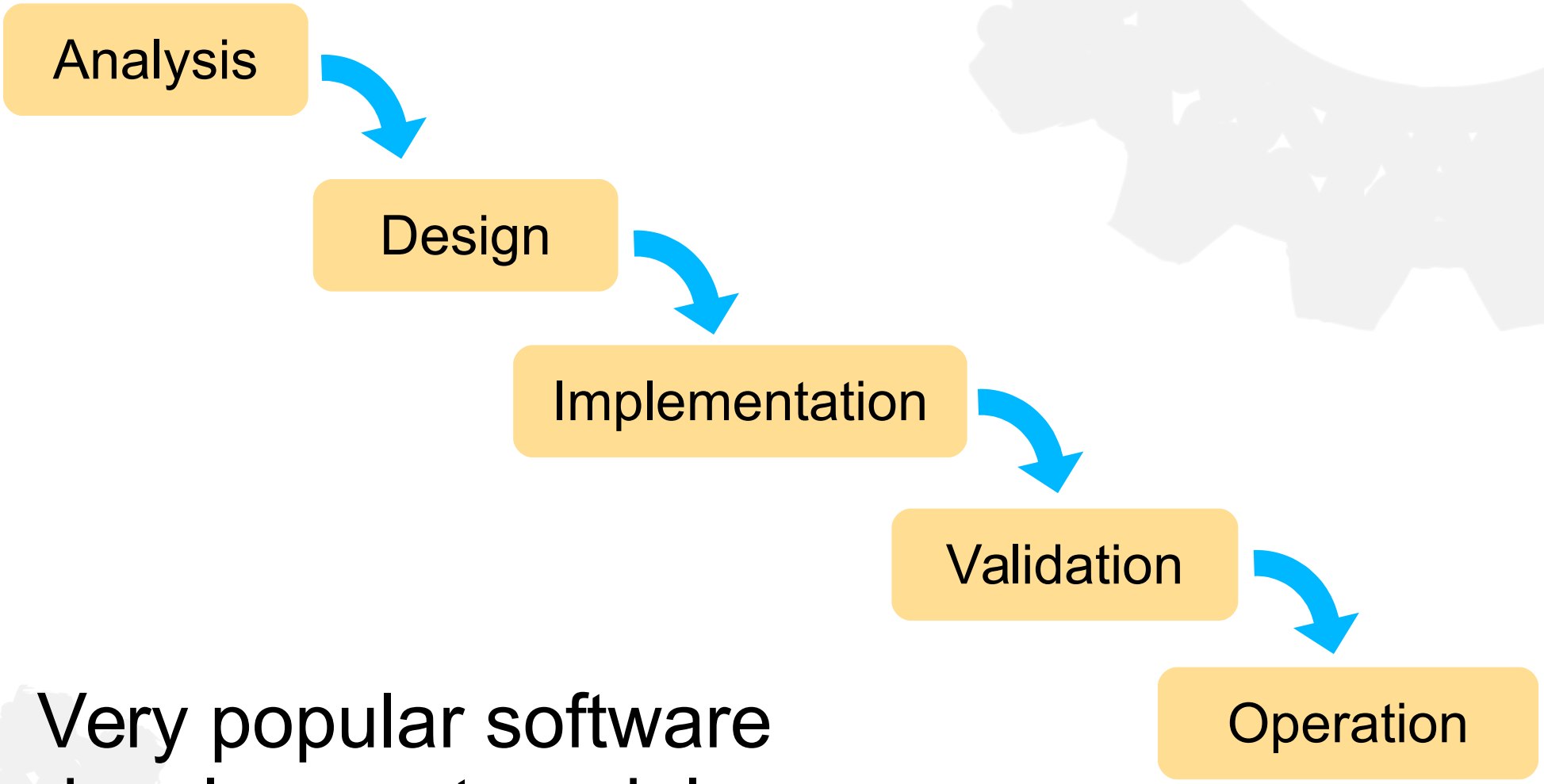


Deployed

We could do better than that

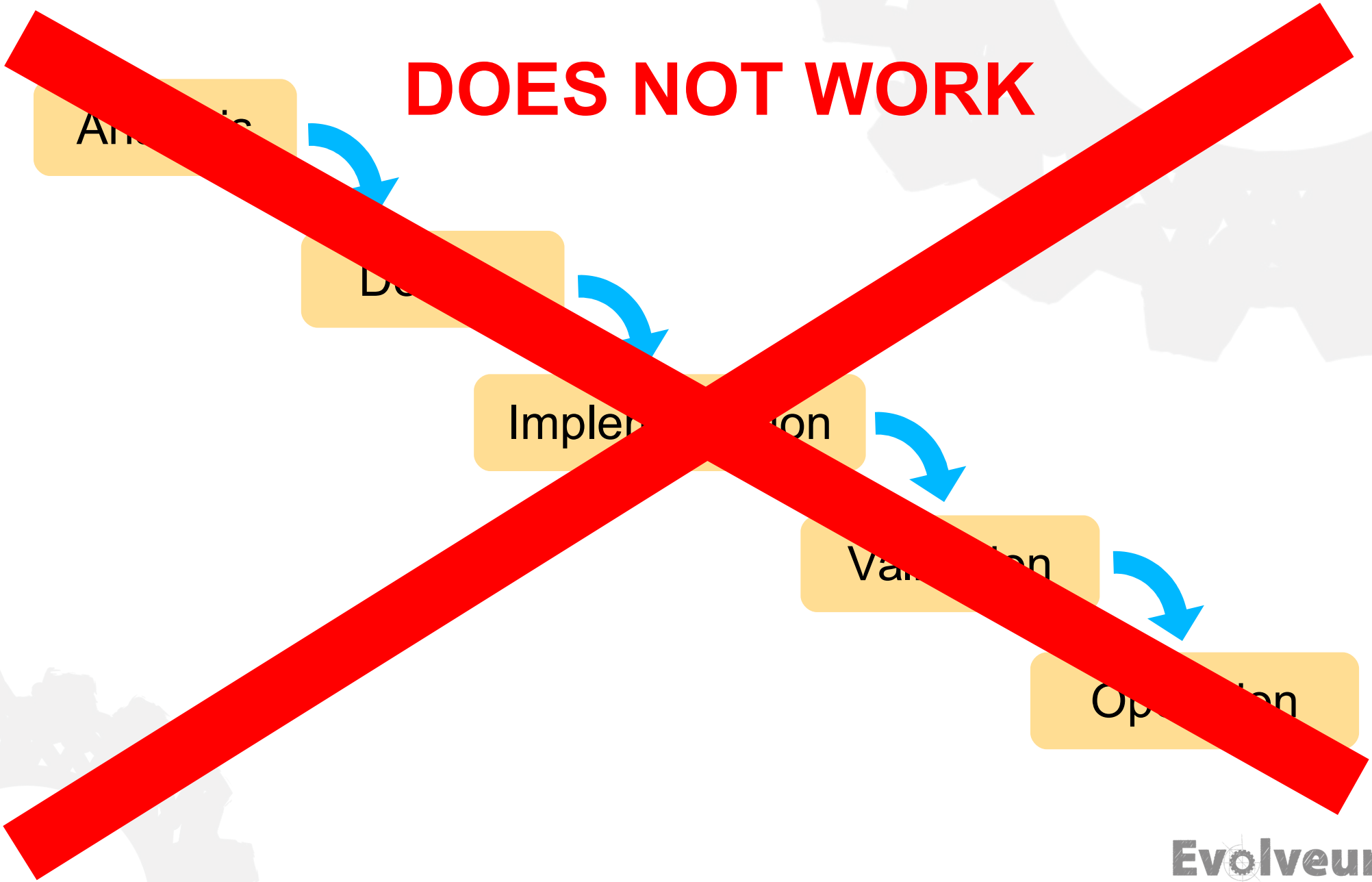
… in theory

Waterfall Model



Very popular software development model

Waterfall Model



Waterfall Model

Hic sunt liones

DOES NOT WORK

High voltage!

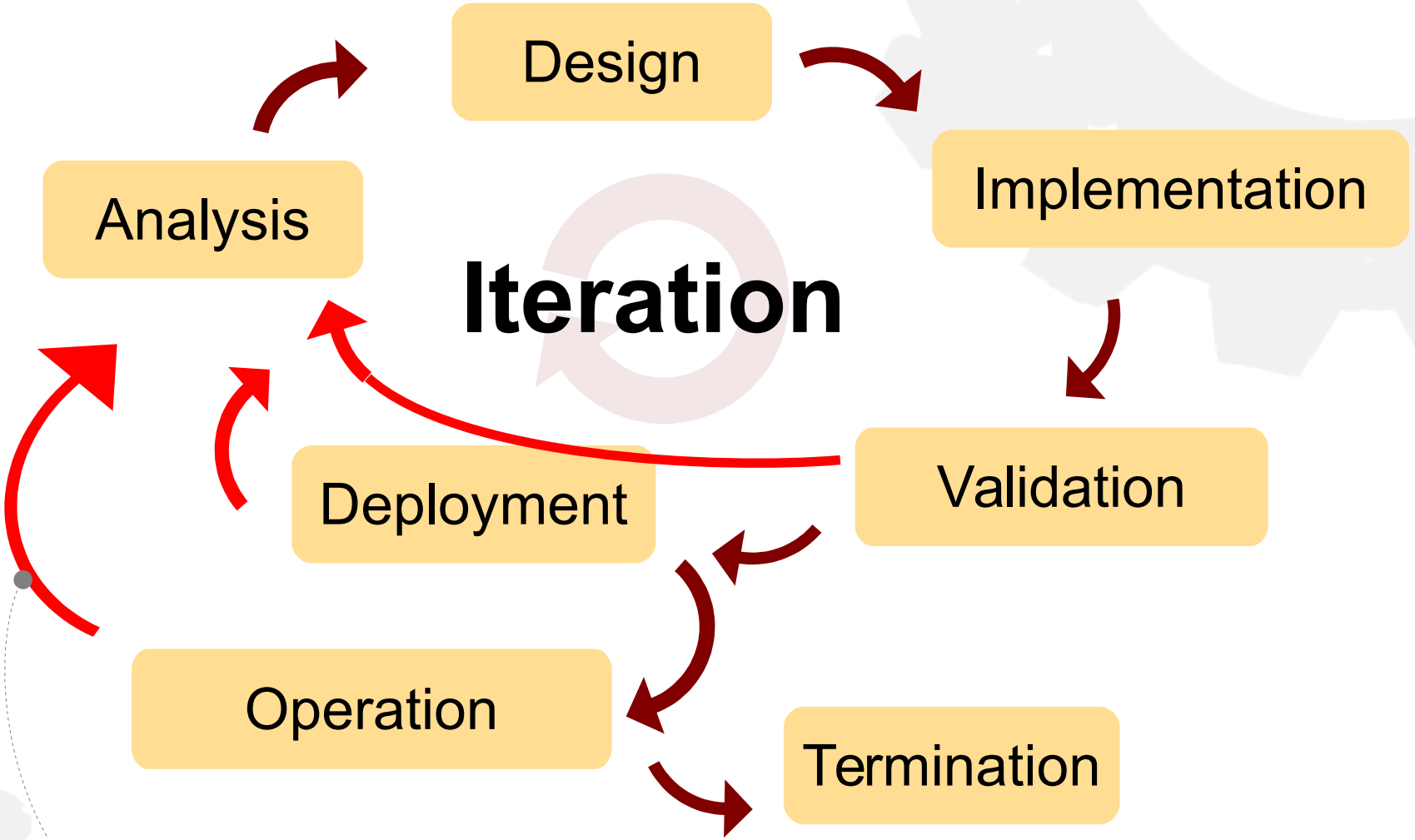
Danger!!!

DO NOT USE

Beware of the Leopard

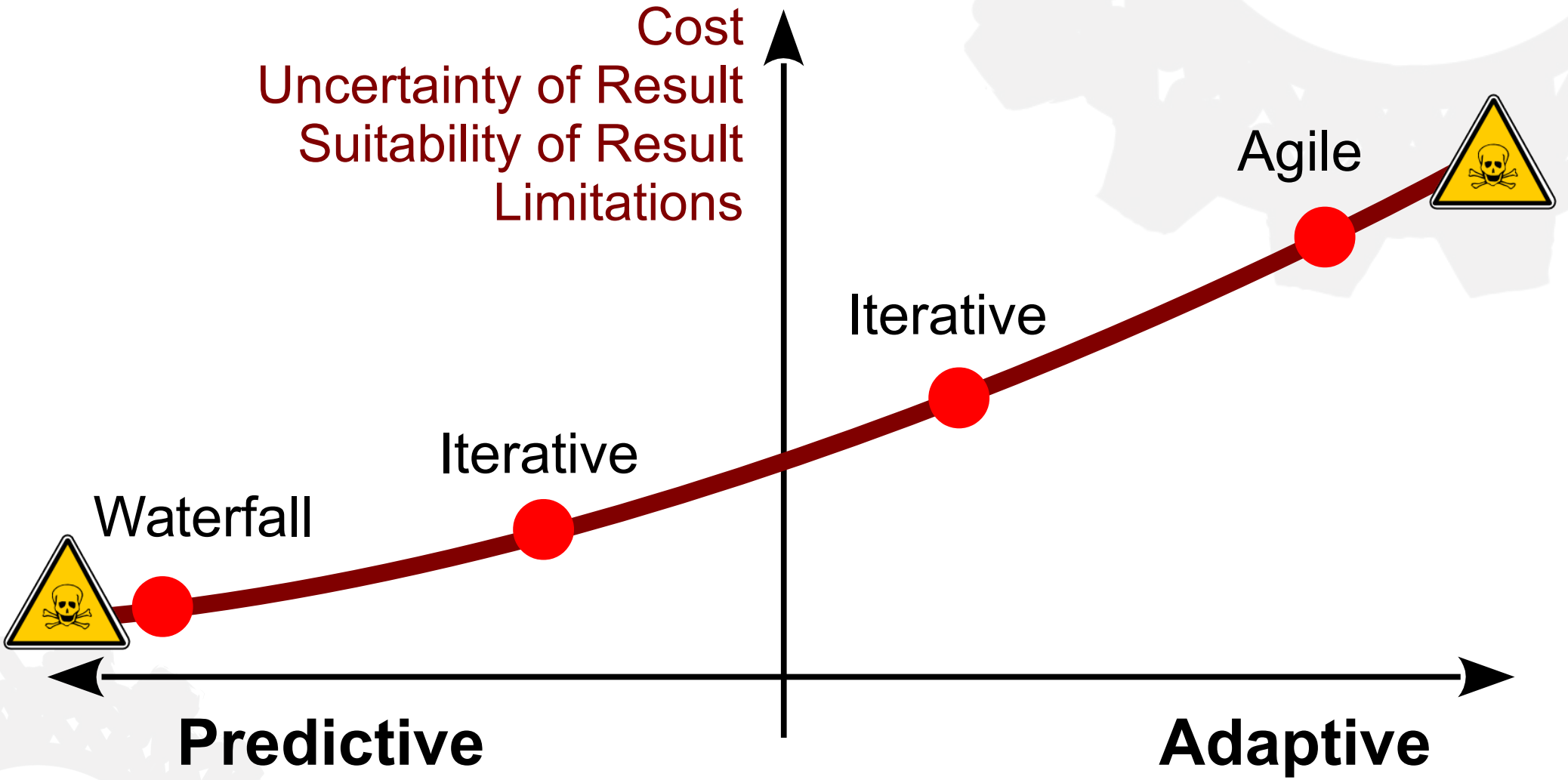
Evolveum

Iterative Development



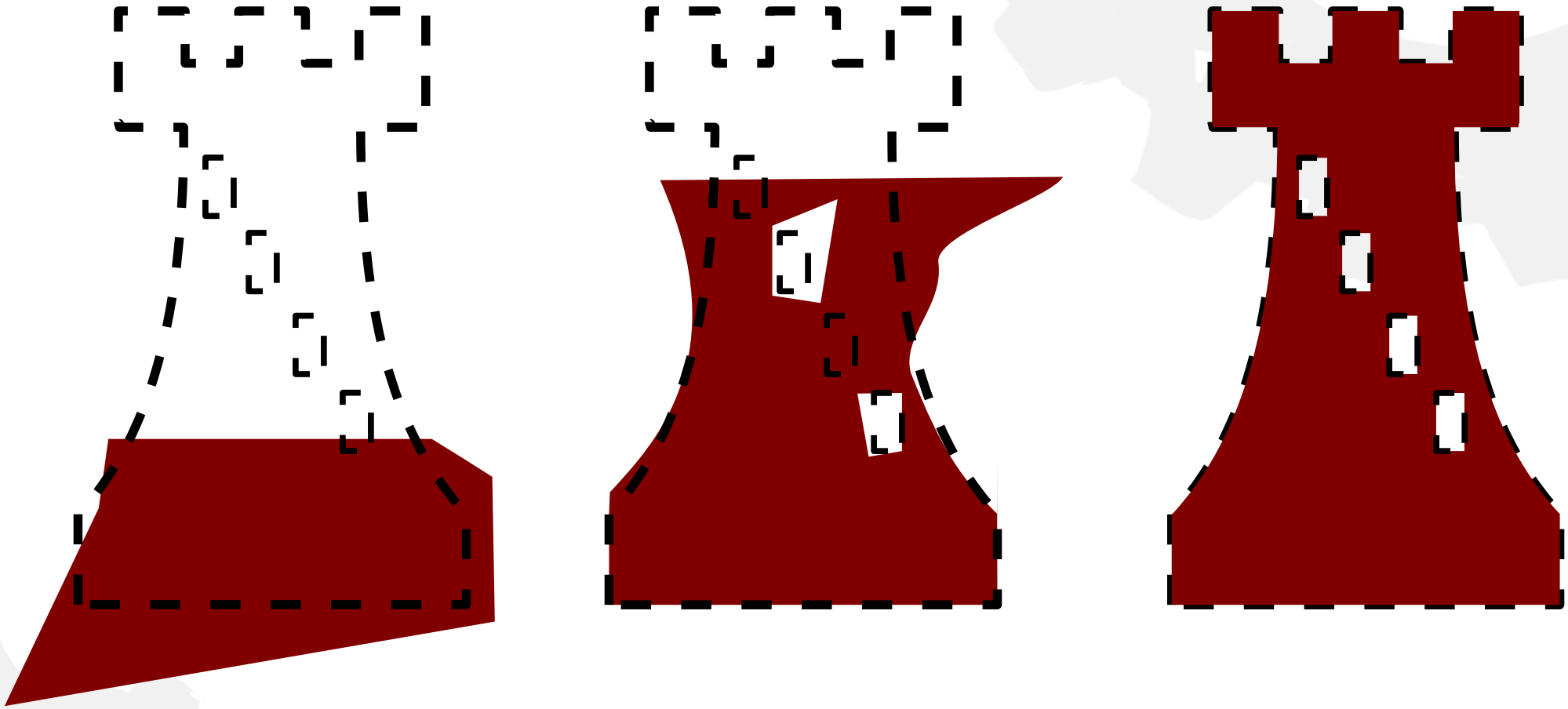
- Feedback
  - Use knowledge gained in previous iteration

Development Methods Summary



We could do better … even in practice

Iterations and Increments



Software Developement … in practice

- Do not try to design/implement everything
  - **Waterfall does not work!**
- Iterations and increments
  - But you need to have some idea about the desired result
- Beware the limitations
  - One size does not fit all
  - Agile does not **always** work
  - Golden hammer (anti-pattern)

Vision

Architecture and design

… in theory

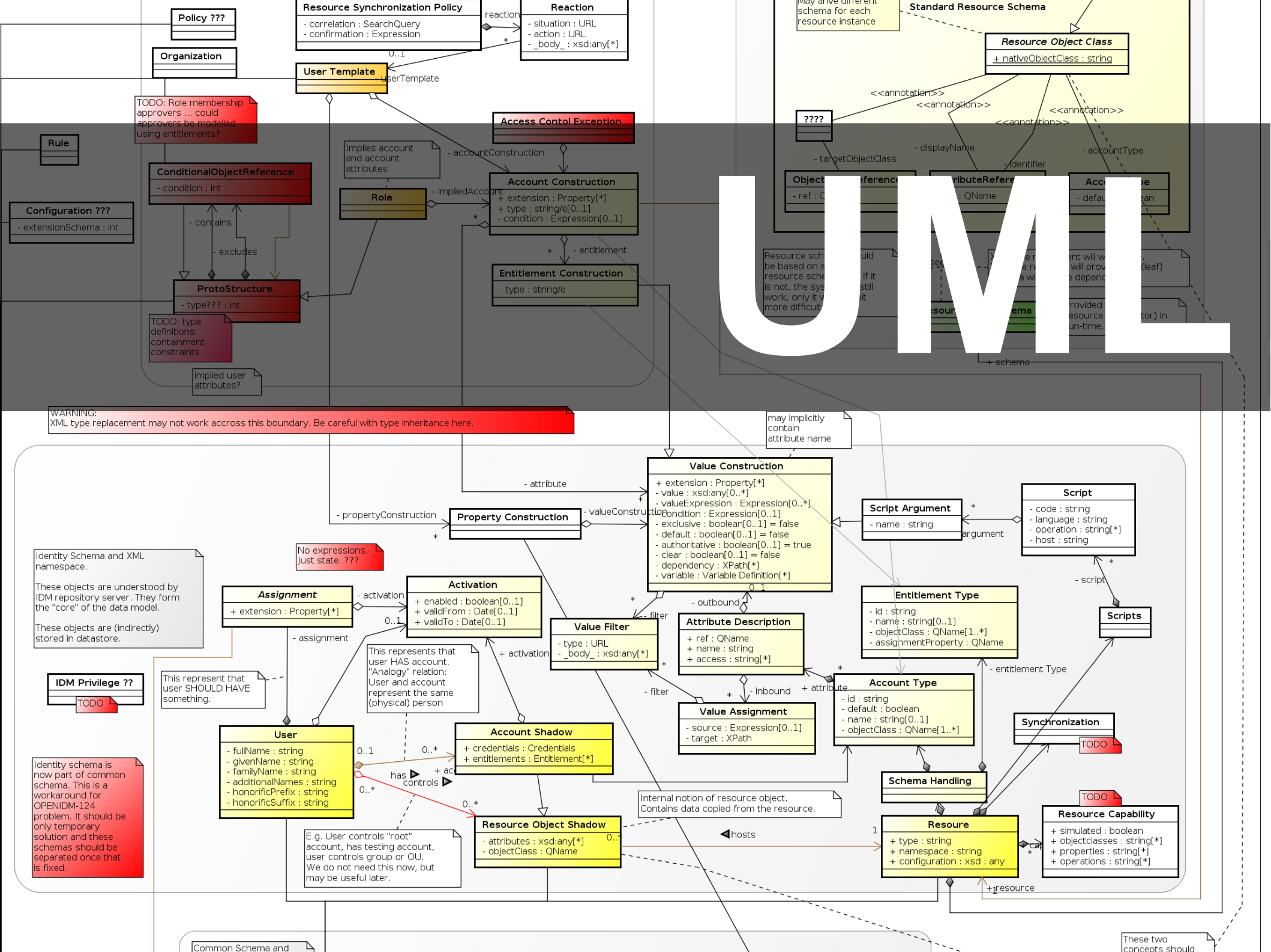
Model

Simplified

= imprecise

Overview

= better “handling”


UML
Policy ???
Organization
Resource Synchronization Policy
- correlation : SearchQuery
- confirmation : Expression
reaction
Reaction
- situation : URL
- action : URL
- _body_ : xsd:any[*]
User Template
- userTemplate
TODO: Role membership approvers ... could approvers be modelled using entitlements?
Rule
ConditionalObjectReference
- condition : int
Configuration ???
- extensionSchema : int
- contains
- excludes
ProtoStructure
- type??? : int
TODO: type definitions: containment constraints
implied user attributes?
Implies account and account attributes
Role
- impliedAccount
Access Contol Exception
- accountConstruction
Account Construction
+ extension : Property[*]
+ type : string/e[0..1]
- condition : Expression[0..1]
- entitlement
Entitlement Construction
- type : string/e
Standard Resource Schema
Resource Object Class
+ nativeObjectClass : string
<<annotation>>
????
- targetObjectClass
- displayName
- identifier
- accountType
+ schema
WARNING:
XML type replacement may not work accross this boundary. Be careful with type inheritance here.
may implicitly contain attribute name
Value Construction
+ extension : Property[*]
- value : xsd:any[0..*]
- valueExpression : Expression[0..*]
- condition : Expression[0..1]
- exclusive : boolean[0..1] = false
- default : boolean[0..1] = false
- authoritative : boolean[0..1] = true
- clear : boolean[0..1] = false
- dependency : XPath[*]
- variable : Variable Definition[*]
- attribute
- propertyConstruction
Property Construction
- valueConstruction
Script Argument
- name : string
- argument
Script
- code : string
- language : string
- operation : string[*]
- host : string
- script
Scripts
Identity Schema and XML namespace.
These objects are understood by IDM repository server. They form the "core" of the data model.
These objects are (indirectly) stored in datastore.
No expressions. Just state. ???
Assignment
+ extension : Property[*]
- activation
- assignment
Activation
+ enabled : boolean[0..1]
+ validFrom : Date[0..1]
+ validTo : Date[0..1]
+ activation
Value Filter
- type : URL
- _body_ : xsd:any[*]
- filter
Attribute Description
+ ref : QName
+ name : string
+ access : string[*]
- outbound
- inbound
- filter
Value Assignment
- source : Expression[0..1]
- target : XPath
Entitlement Type
- id : string
- name : string[0..1]
- objectClass : QName[1..*]
- assignmentProperty : QName
- entitlement Type
Account Type
- id : string
- default : boolean
- name : string[0..1]
- objectClass : QName[1..*]
+ attribute
IDM Privilege ??
TODO
This represent that user SHOULD HAVE something.
This represents that user HAS account. "Analogy" relation: User and account represent the same (physical) person
User
- fullName : string
- givenName : string
- familyName : string
- additionalNames : string
- honorificPrefix : string
- honorificSuffix : string
has
controls
Account Shadow
+ credentials : Credentials
+ entitlements : Entitlement[*]
Resource Object Shadow
- attributes : xsd:any[*]
- objectClass : QName
hosts
Internal notion of resource object. Contains data copied from the resource.
Schema Handling
Synchronization
TODO
Resoure
+ type : string
+ namespace : string
+ configuration : xsd : any
+resource
TODO
Resource Capability
+ simulated : boolean
+ objectclasses : string[*]
+ properties : string[*]
+ operations : string[*]
Identity schema is now part of common schema. This is a workaround for OPENIDM-124 problem. It should be only temporary solution and these schemas should be separated once that is fixed.
E.g. User controls "root" account, has testing account, user controls group or OU. We do not need this now, but may be useful later.
Common Schema and
These two concepts should

Architecture and design

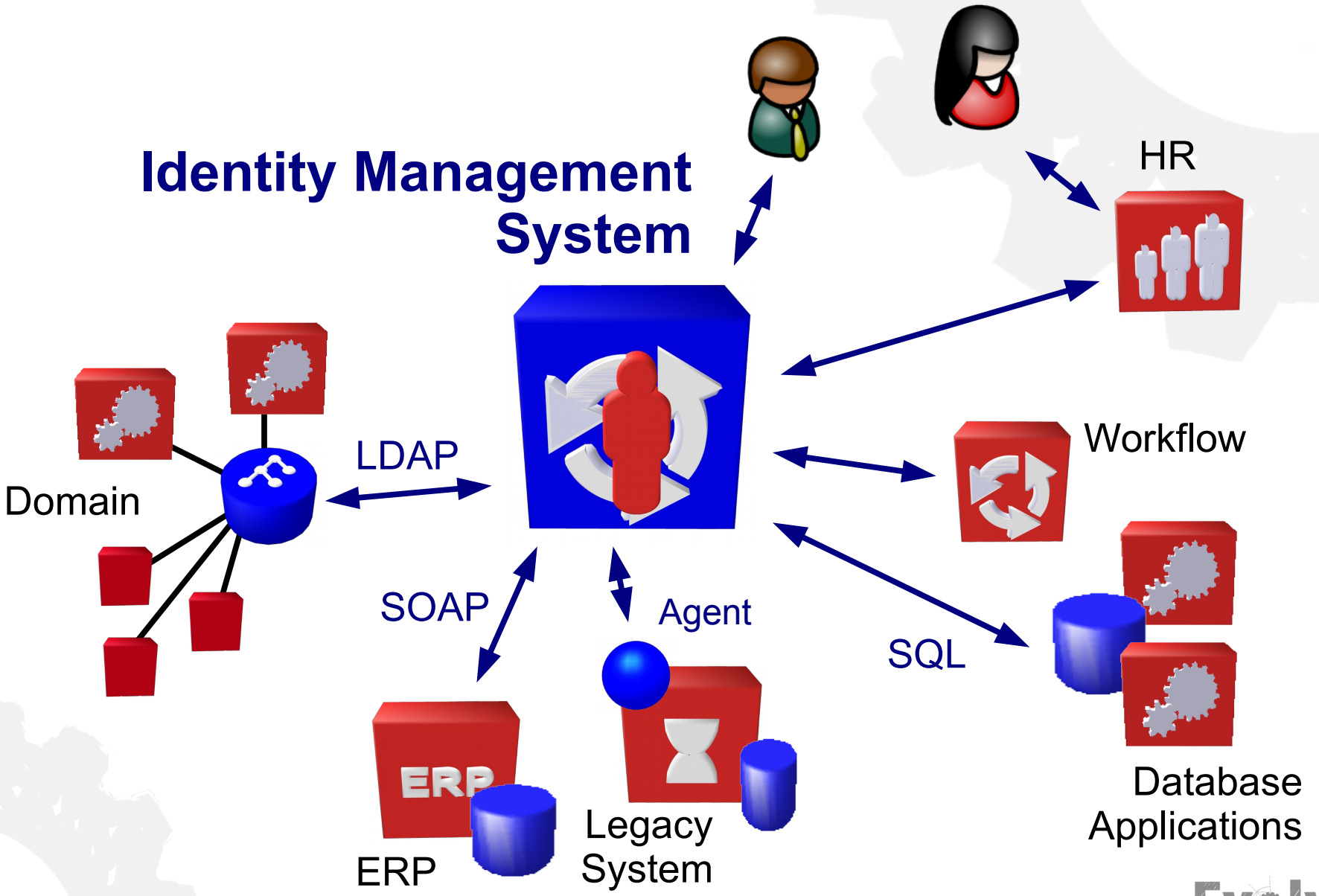
… in practice

Models

- Models in pure form (e.g. pure UML)
  - Limited usefulness
  - Fighting with tools instead of making progress
- Hybrid (customized) models
  - Very useful, especially in early phases
  - Difficult to maintain
- Free-form diagrams
  - Whiteboard – absolutely necessary
  - Brainstorming, early “validation”

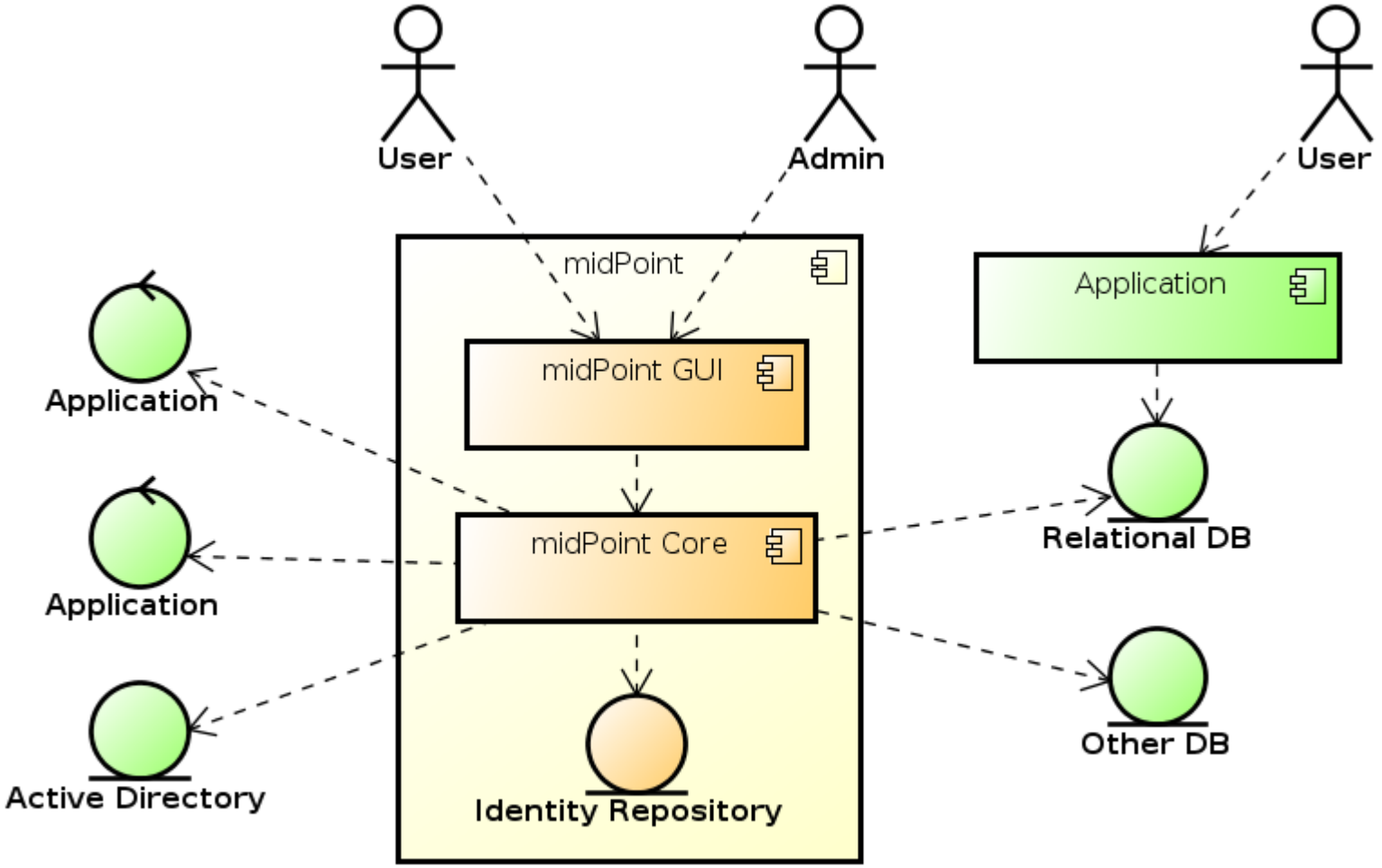
Informal Architecture Diagram

(Technical marketing)

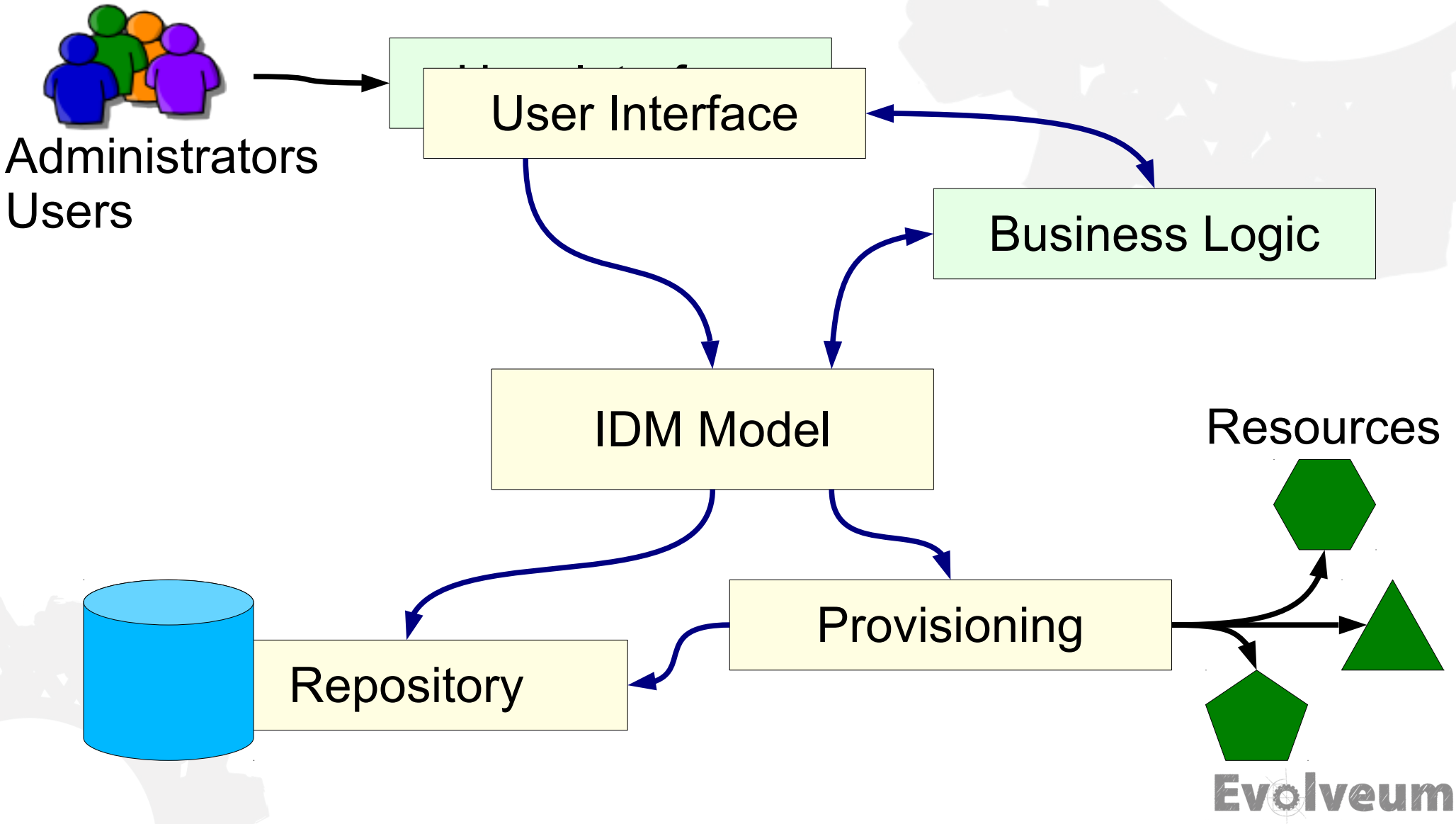


Formal Architecture Diagram

(I have UML and I'm not afraid to use it)

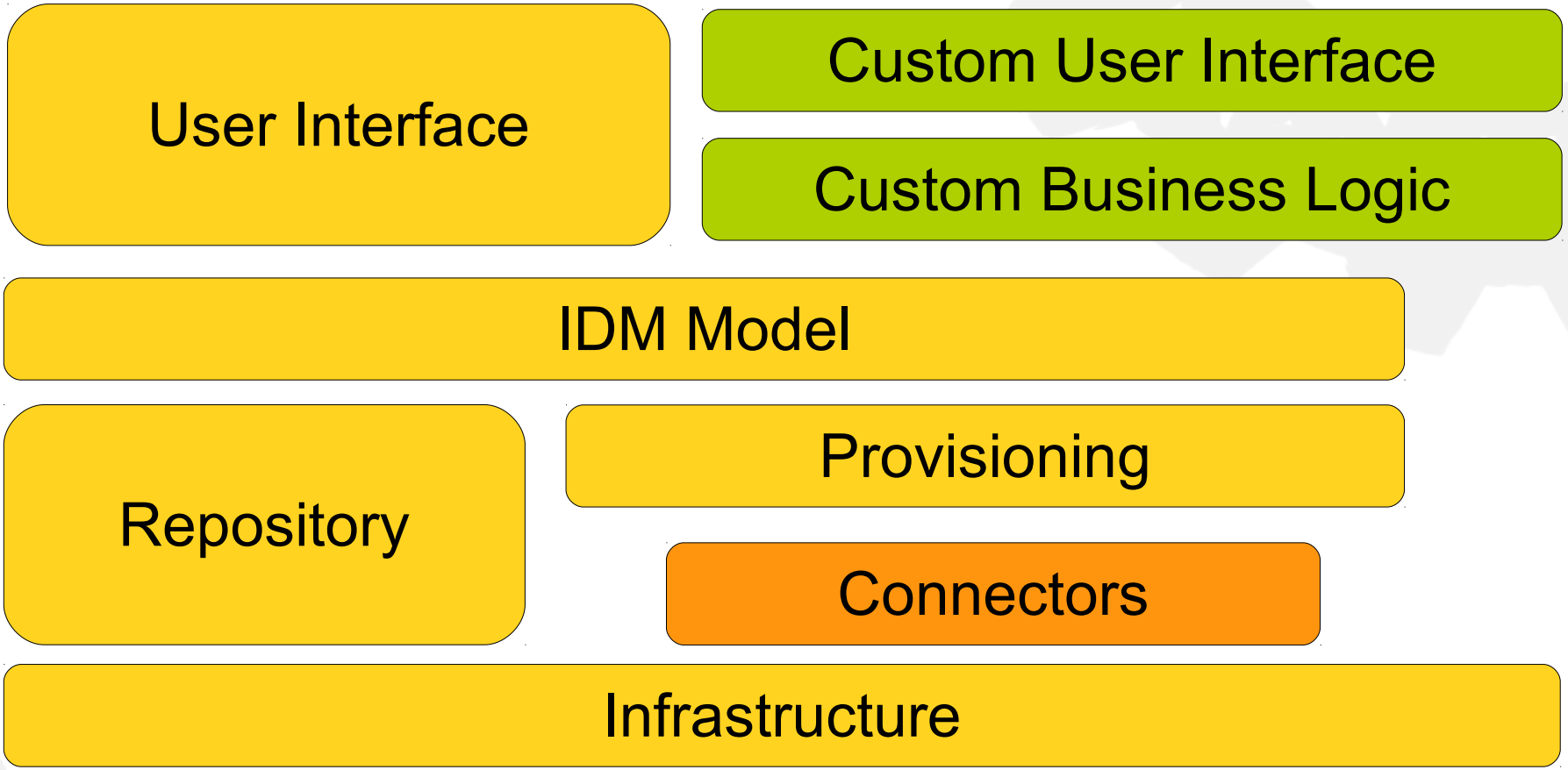


Informal Component Diagram
(Whiteboard 2.0)



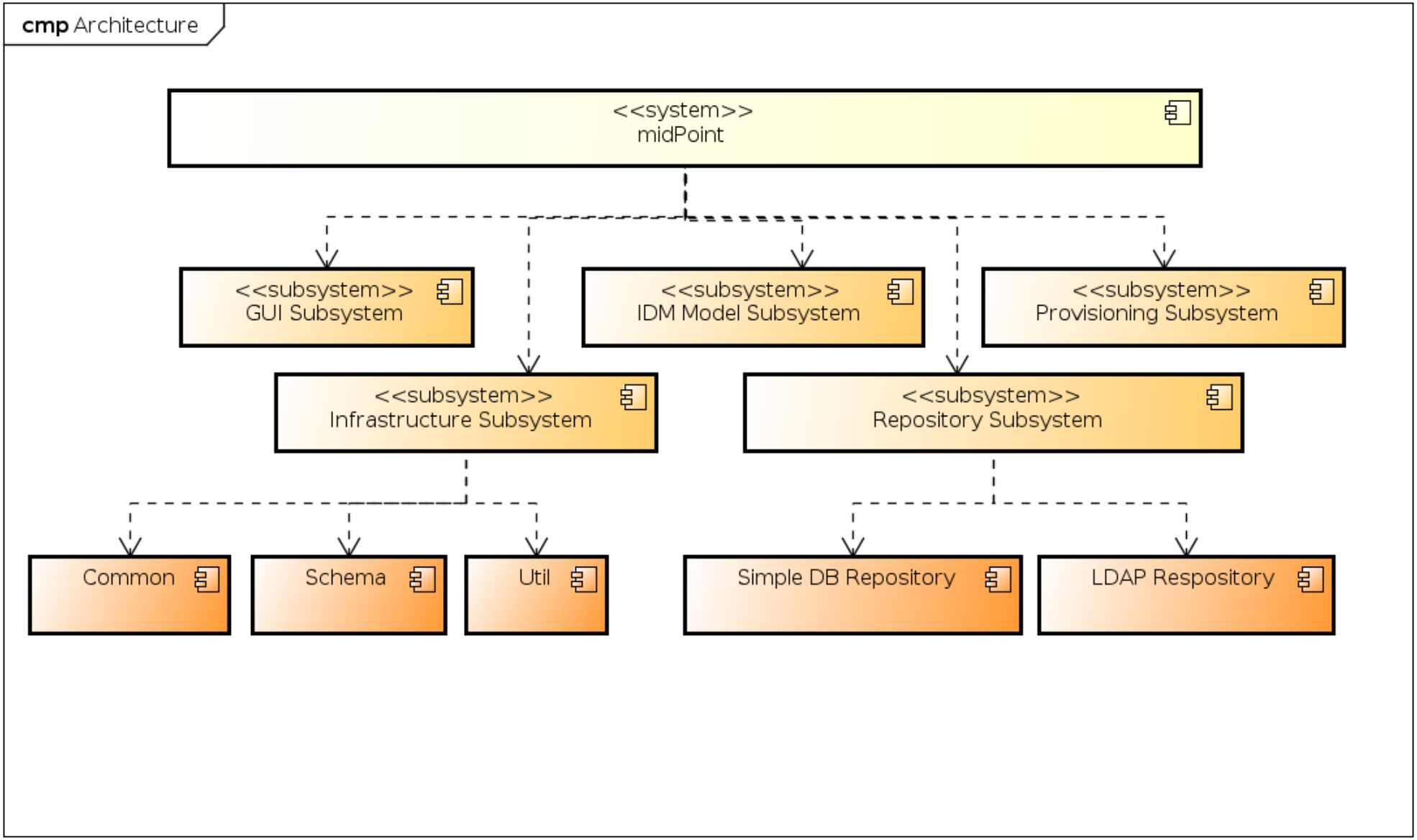
Marketing-Oriented Diagram

(Boxes and more boxes)

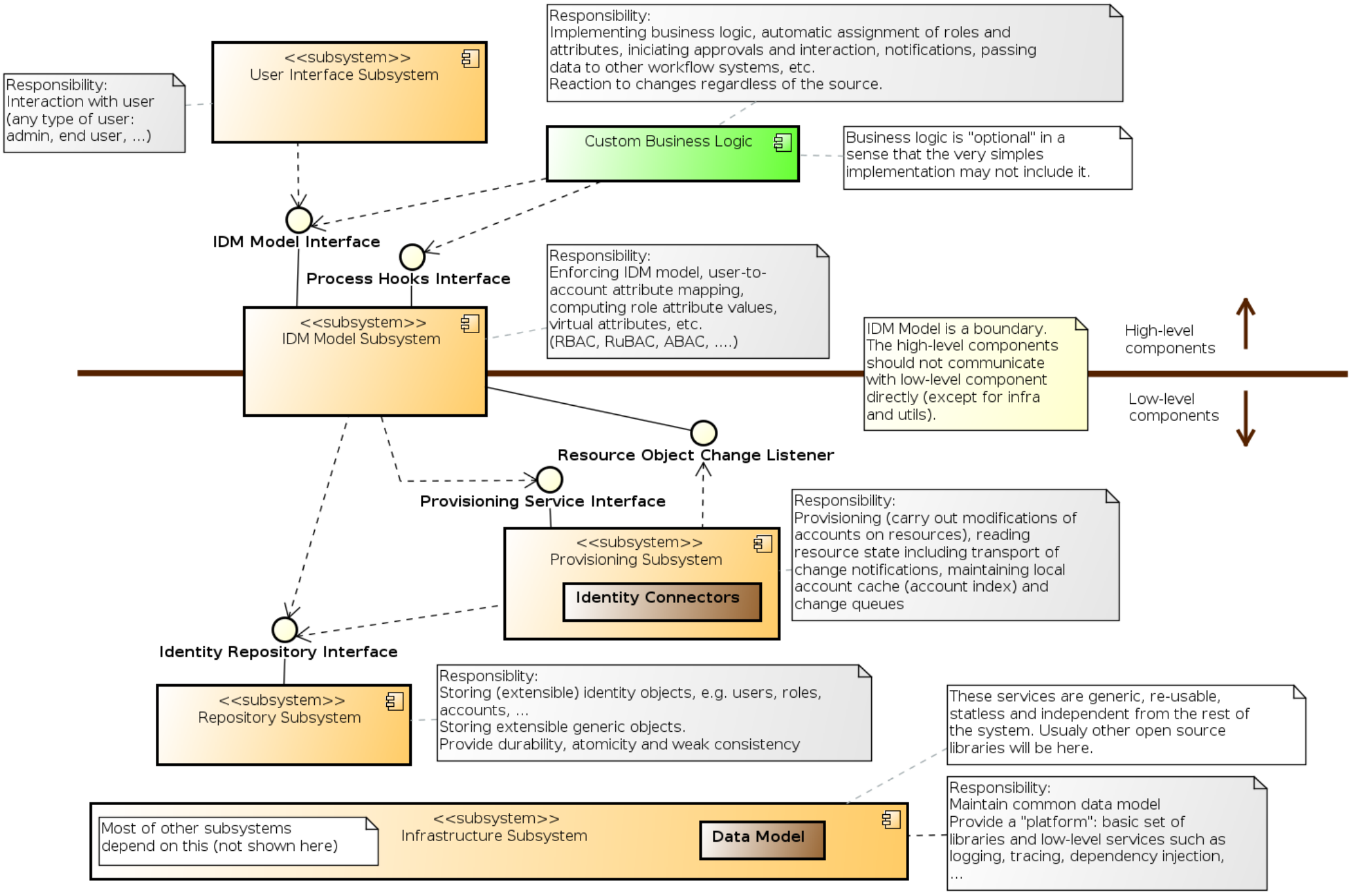


If you ever see this: run away!

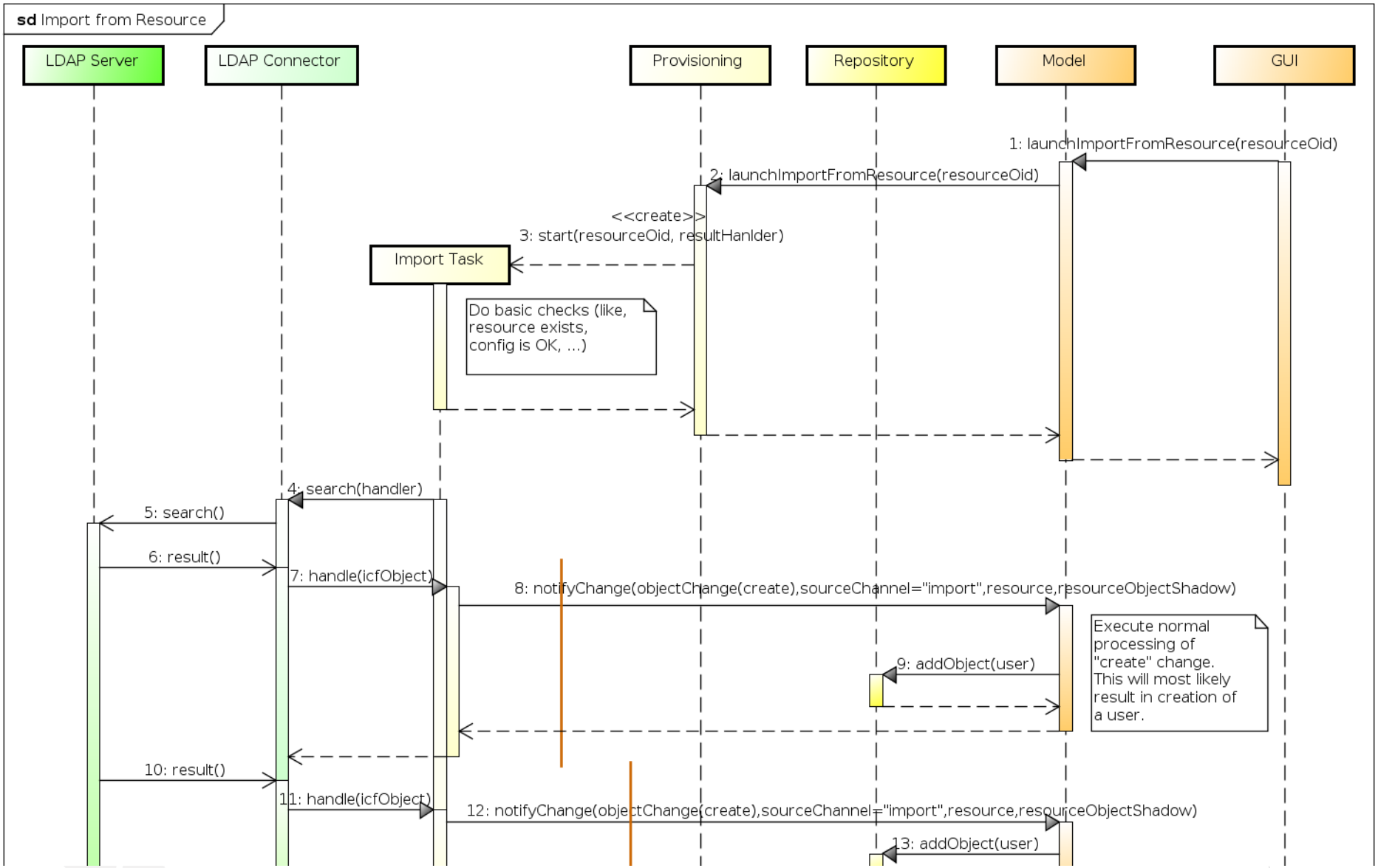
System Decomposition



Modular and Component Structure

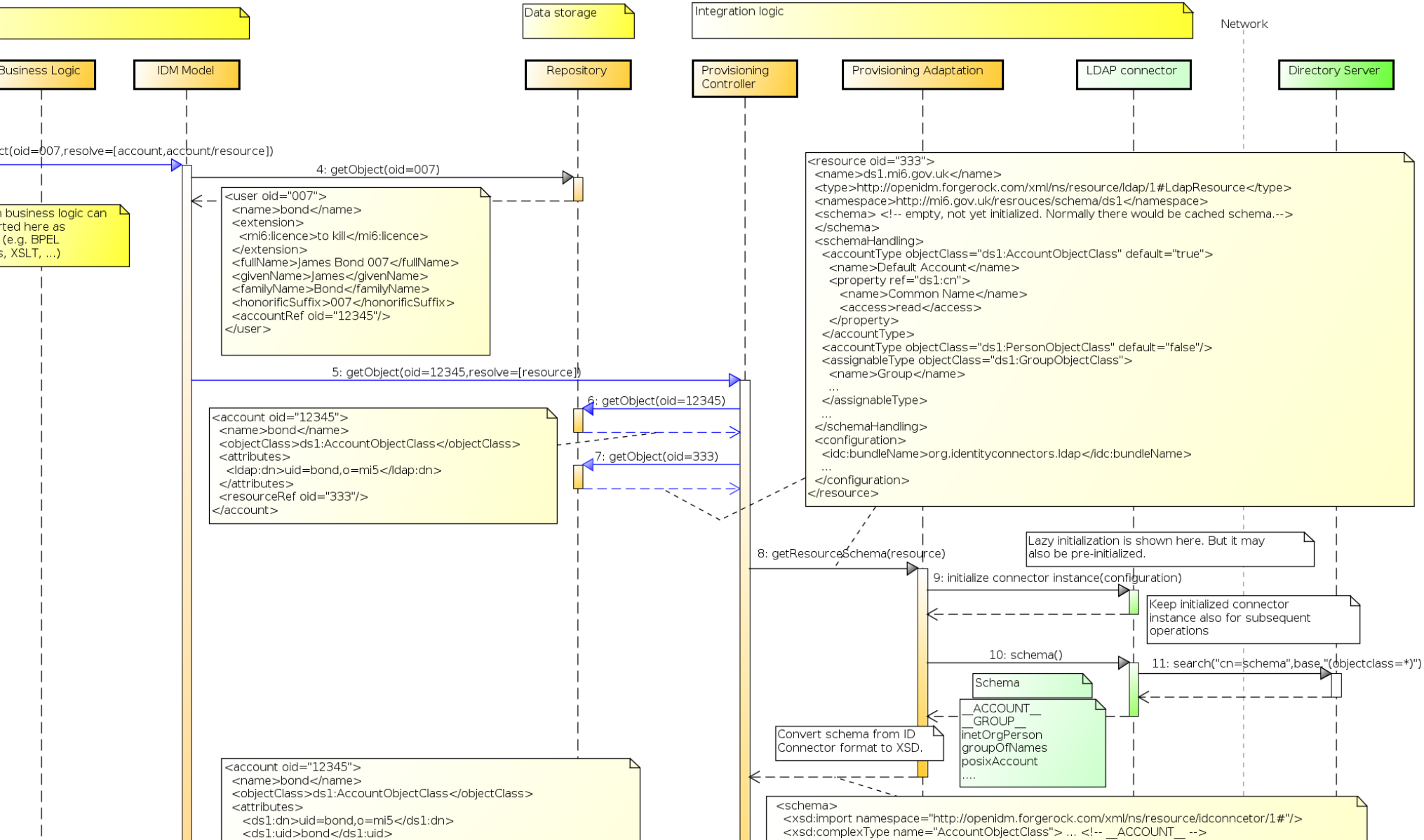


Component Interactions

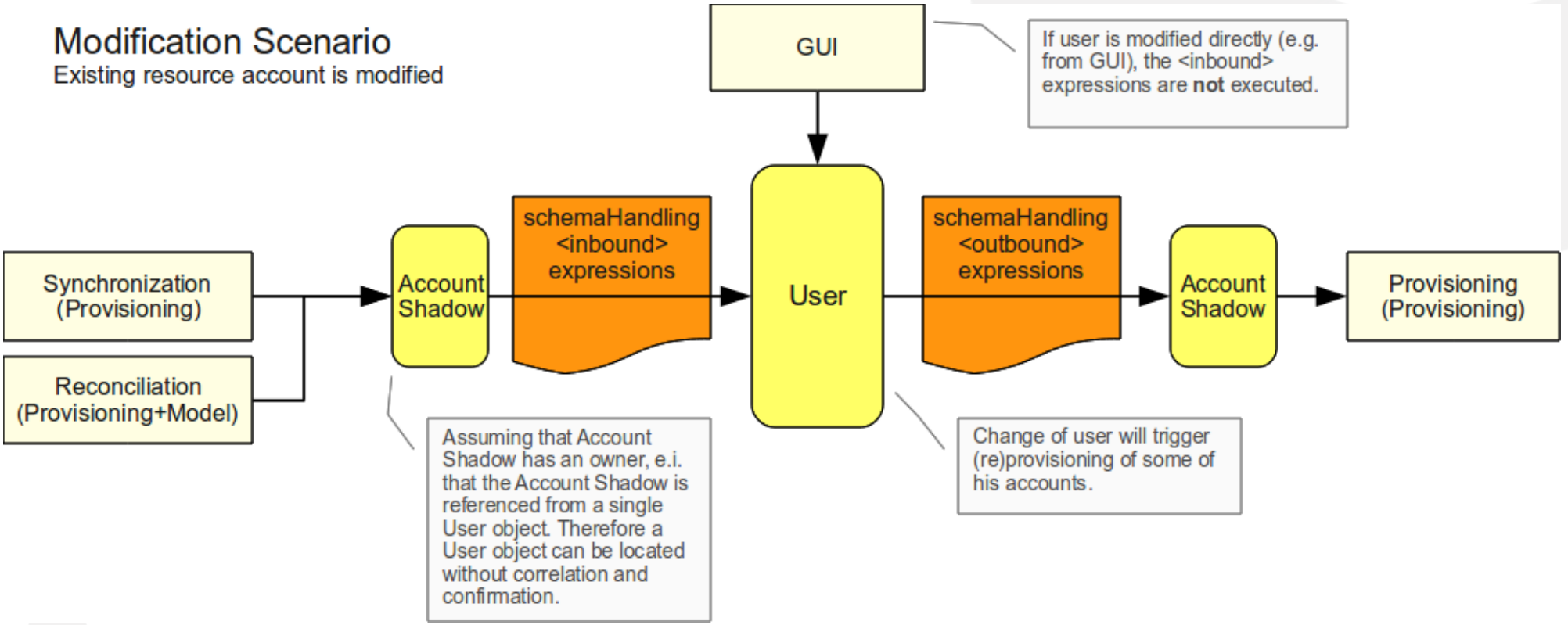


Evolveum

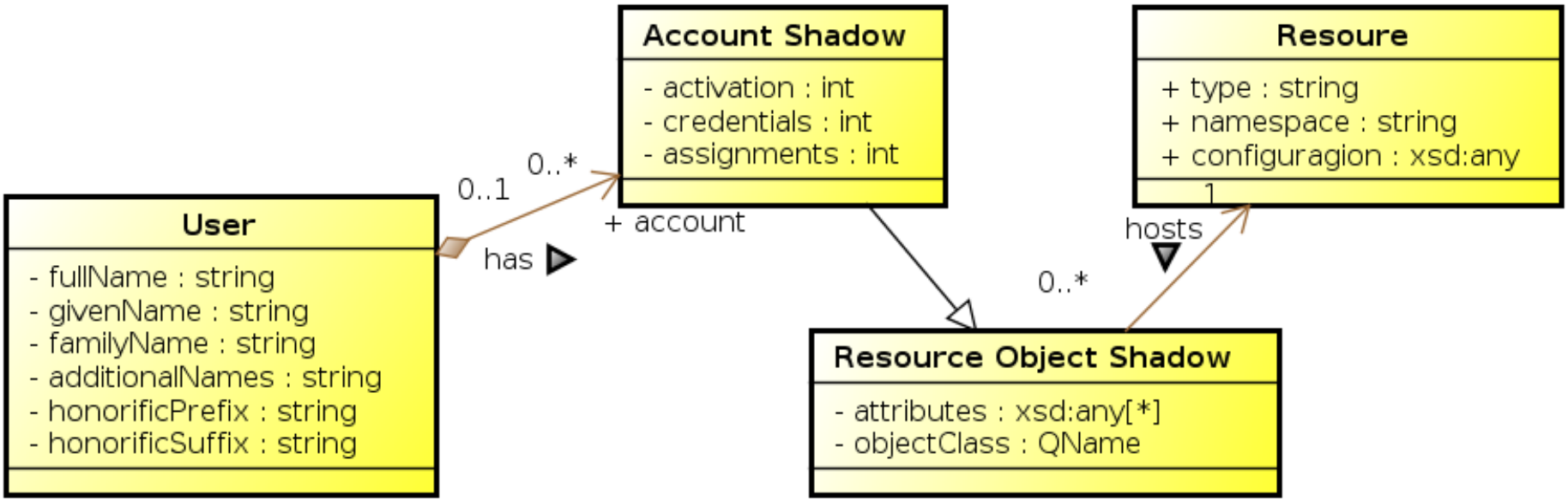
Component Interactions



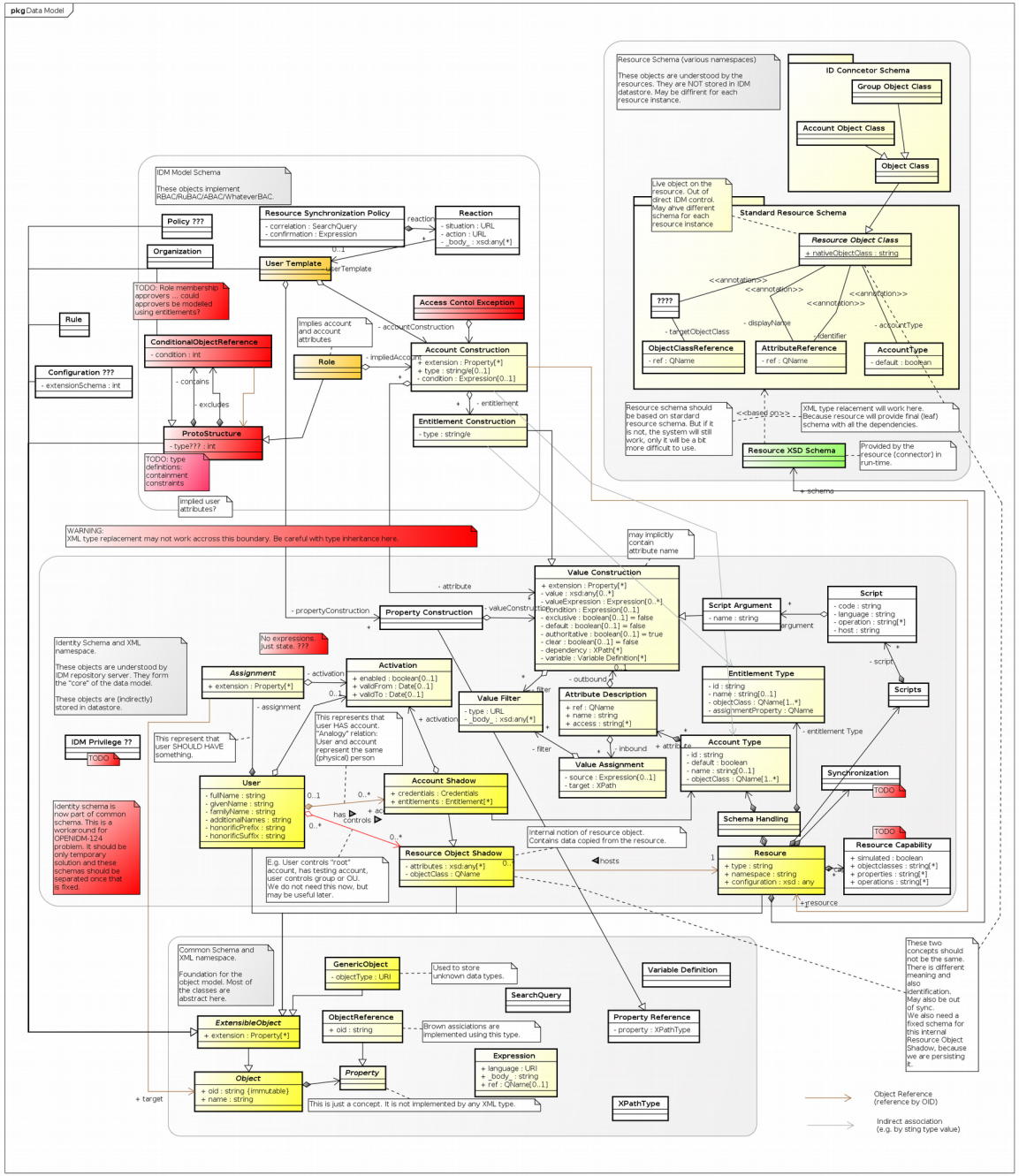
Component Interactions



Data Structures



Complex Data Structures



- Hard to maintain
- Data schema
- Generate?

Evolveum

Architecture Model Summary

- Operates with concepts
  - May or may not map to final components, interfaces, ...
- Difficult to align with implementation
  - ... and not efficient to reach 100% alignment
  - The model should be guideline, not dogma
- Model ≠ Architecture
  - Architecture is much more:
    - Textual descriptions, explanations, description of concepts
    - Motivations, design decisions, trade-offs, future expectations
  - **Beware** of tools that promise to simplify that

WARNING

Architectural Principles

Those are (very) useful

- Separation of concerns
- Dependency inversion principle
- Acyclic dependencies principle
- Stable abstractions principle
- Stable dependencies principle
- Open-closed principle
- Single responsibility principle
- Interface segregation principle
- …

When architecture goes wrong …

Fallacies, Antipatterns, Rot & Smell

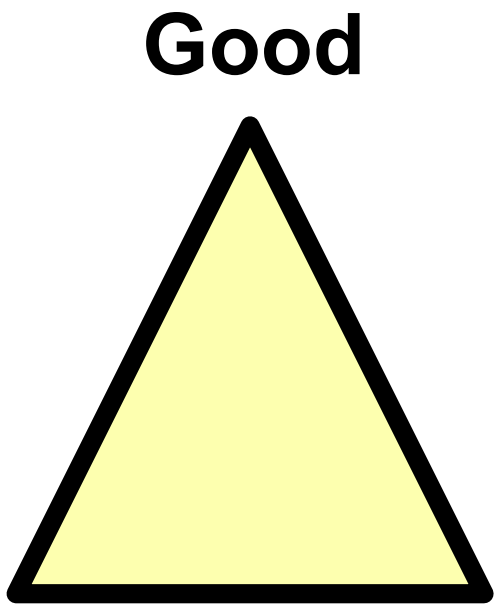
- Fallacies of distributed computing
  - Network is reliable, Latency is zero, Bandwidth is infinite, ...
- Architectural antipatterns
  - Big ball of mud, Design by committee, Not invented here, ...
- Symptoms of rotting design
  - Rigidity, Fragility, Immobility, Viscosity
- Code smell
  - Duplicated code, Contrived complexity, Feature envy, ...

Common Problems

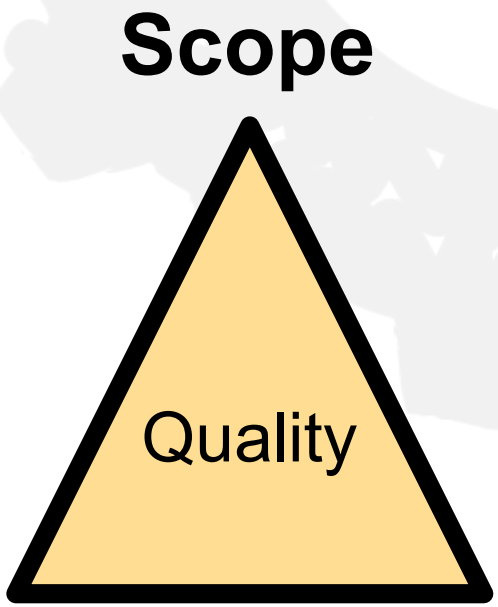
- Too little analysis / design
  - Especially in agile and open source
- Too much architecture ("stratospheric architecture")
  - Pretty concepts that never get implemented
- No environment analysis
- Unmaintained architecture
  - Architect *did his work* at beginning of the project

    … and then left
  - Architecture is a mutable thing! Needs constant maintenance.

Iron Triangle



Pick any two ...
... the third will follow



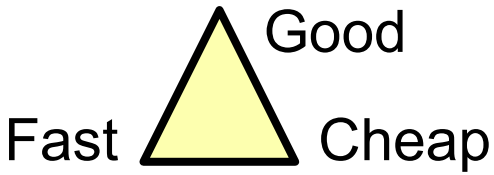
At least one corner must be variable, otherwise quality will suffer

Moving Target

- Requirements are incomplete and changing
- Environment is changing

  => software must change

- Architecture must be able to adapt
- Expect that you will have to make changes
- Do not forget about Iron Triangle

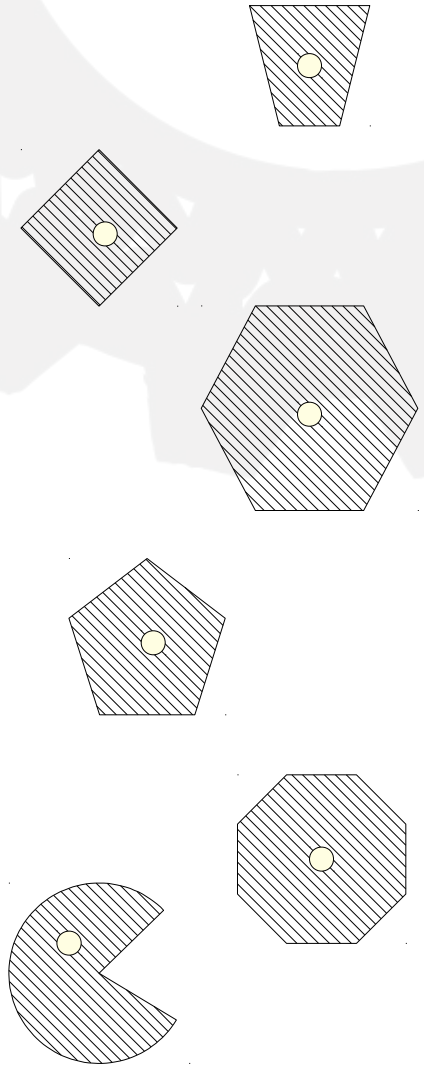


Buzzword-Oriented Architecture

- Very common approach
- Huge problem
- Solution: known what you are doing
  - Understand the technology before committing to it
- History repeating
  - Basic principles do not change often

History Repeating

- 1976: RFC 707
- 1981: Xerox Courier
- 1991: CORBA
- 1993: DCE/RPC → DCOM
- 1995: SunRPC
- 1998: SOAP
- 200x: “RESTful” API



What we can do?

Form follows purpose

Form Follows Purpose

versus

Pragmatic Approach

- Focus on the effects of the architecture
  - Emphasize the aspects that can help achieve results
  - Ignore aspects that does not influence result
- Common sense, simplicity
- Continuous change
- Skepticism
  - Continual testing, systematic doubt
  - True knowledge is uncertain

Questions and Answers

Thank You

Radovan Semančík

www.evolveum.com