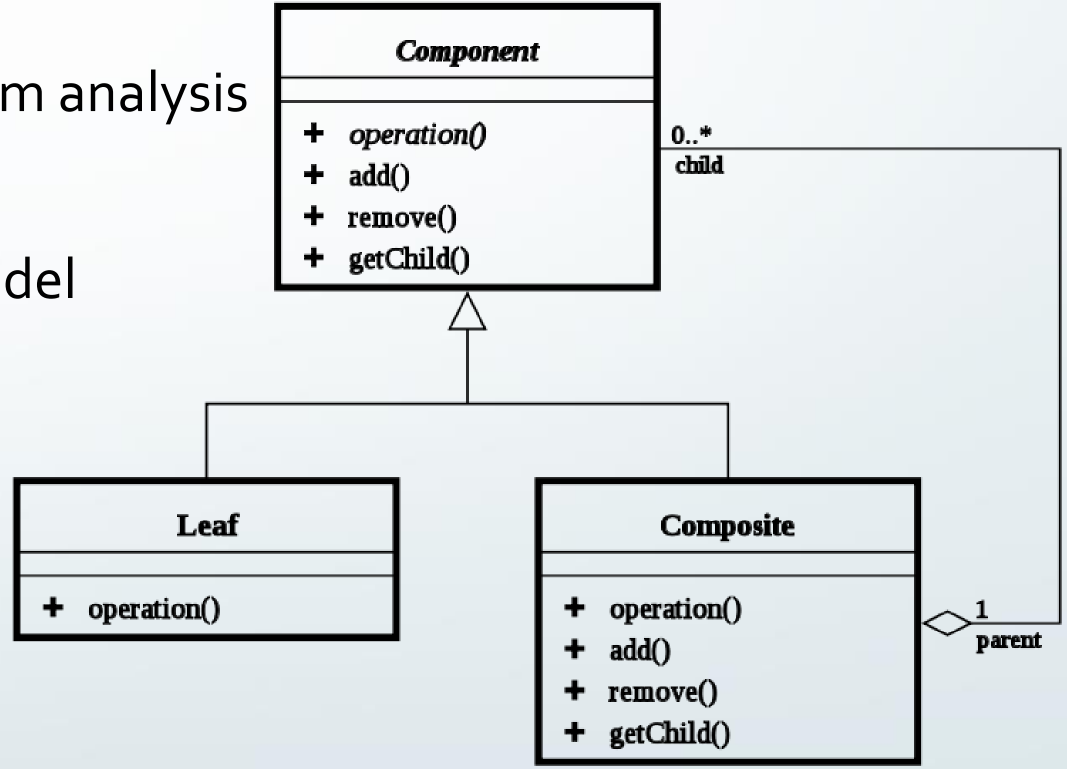
Generating UML Models with Inferred Types from Pharo Code

JAN BLIZNIČENKO AND ROBERT PERGL

CZECH TECHNICAL UNIVERSITY IN PRAGUE

Models and diagrams

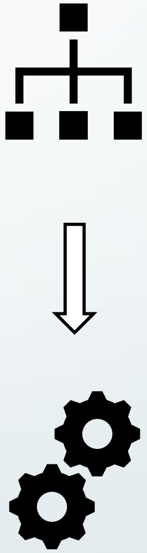
- Models and diagrams help with legacy system analysis
- Model is an abstraction of the system
- Diagram is a visual representation of the model
- Most common: UML
  - Widely used tools:
    Enterprise Architect,
    Modelio
    and many other



Source: Wikimedia Commons

Keeping UML models up-to-date

- Updating models, generating code from model
- Updating models and code together
- Updating models based on code changes



Keeping UML models up-to-date

- Models are often outdated
- … or there are no models at all
- What to do now?

How to get UML model from code?

- Lots of manual work

  or

- Generating

UML related tools in Pharo

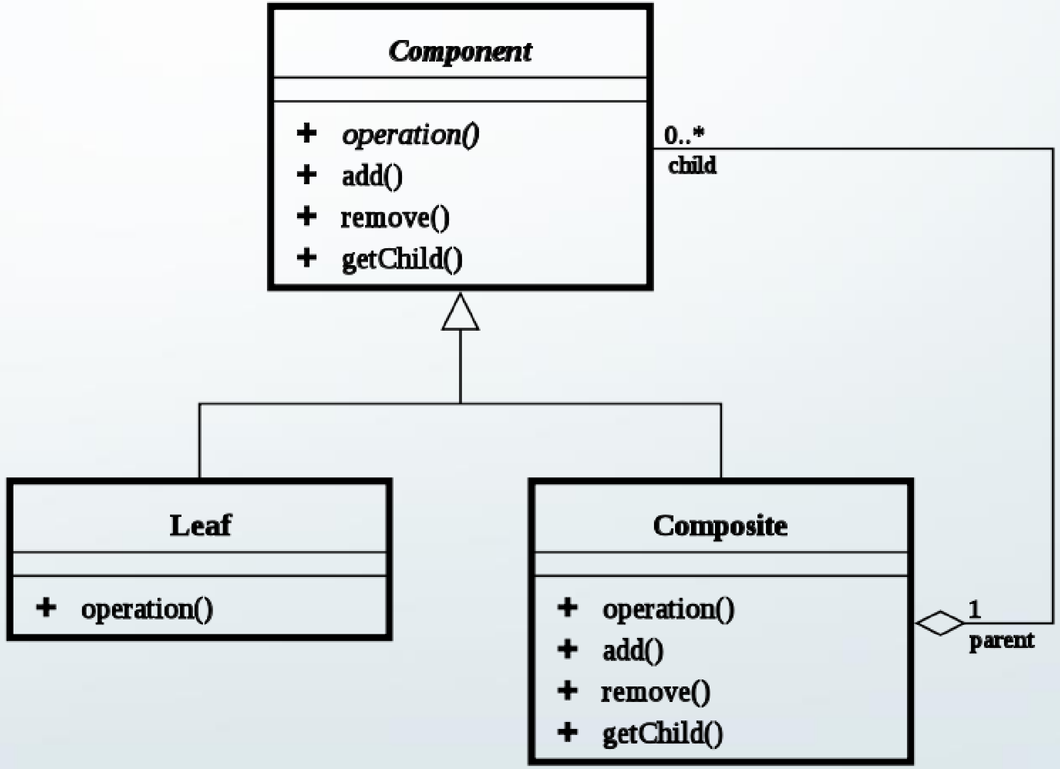
- GraphViz binding
  - generating UML class diagrams, no models, no associations
- Moose with FAMIX models
  - generating models similar to UML, uses RoelTyper to find types
- OpenPonk
  - creating UML class structure models (actual UML by specification), no generating

Need for types

Types

⇩

dependencies between classes or packages



Source: Wikimedia Commons

Type inference

- Static – analysis of the code itself / AST:
  - assignments of just-created instances
  - messages send expressions
- Real-time / dynamic – logging types of running code
  - requires running application, or thorough tests or examples

Type inference tools in Pharo

- RoelTyper – variables only, no contents of collections
- RBRefactoryTyper – variables only, includes contents of collections
- J2Inferer – variables, method parameters and return types
- Several research results regarding real-time type inference

Goal

- Long term goal: Generating UML Diagrams from Smalltalk code and using them in broadly used tools like Enterprise Architect
- For now: focusing specifically on generating structural UML models from Pharo code.
  - Models for Class diagrams

Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Diagrams with as many automation as possible?

Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Class Diagrams with as many automation as possible?

How to represent UML models in Pharo?

- UML metamodel for OpenPonk
  - Independent from OpenPonk
  - Generated from UML specification

Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Class Diagrams with as many automation as possible?

How to generate a UML class structure model?

- Moose has Pharo to FAMIX importer
  - FAMIX to UML convertor could be created
  - Advantage: reusing existing project
- Custom generator inspired by FAMIX importer
  - Advantage: independent on FAMIX and Moose, single step instead of two

How to generate a UML class structure model?

- Moose has Pharo to FAMIX importer
  - FAMIX to UML convertor could be created
  - Advantage: reusing existing project
- Custom generator inspired by FAMIX importer
  - Advantage: independent on FAMIX and Moose, single step instead of two

Structure generator

Includes:

- classes of the package with all details
- classes the package depends on
  - without details – customizable
  - without their dependencies

Dependencies of the package

- Superclasses
- Classes of objects used by classes in the package
  - Object assigned to an instance/shared variable
  - Object passed as an argument
  - Object returned
  - ~~Object used temporarily in a single method~~

Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Class Diagrams with as many automation as possible?

How to find types of instance variables, method parameters and return types?

- TypesManager handles finding data types

TypesManager

- In case the type is already known, return it
    - found types are persistent
- In case the type is not known, find it!

How should TypesManager find the type?

- Use any of the static type inferrers or real-time type inference?
- Try to do it automatically or just help the user (analyst)?

How should TypesManager find the type?

- Adaptors for type inferrers
  - Possibility to create custom inferrer or use any other one
- Type inferrers combinator
  - If one inferrer did not find a type, the other one might
- These return possible types
  - One, multiple ones or none

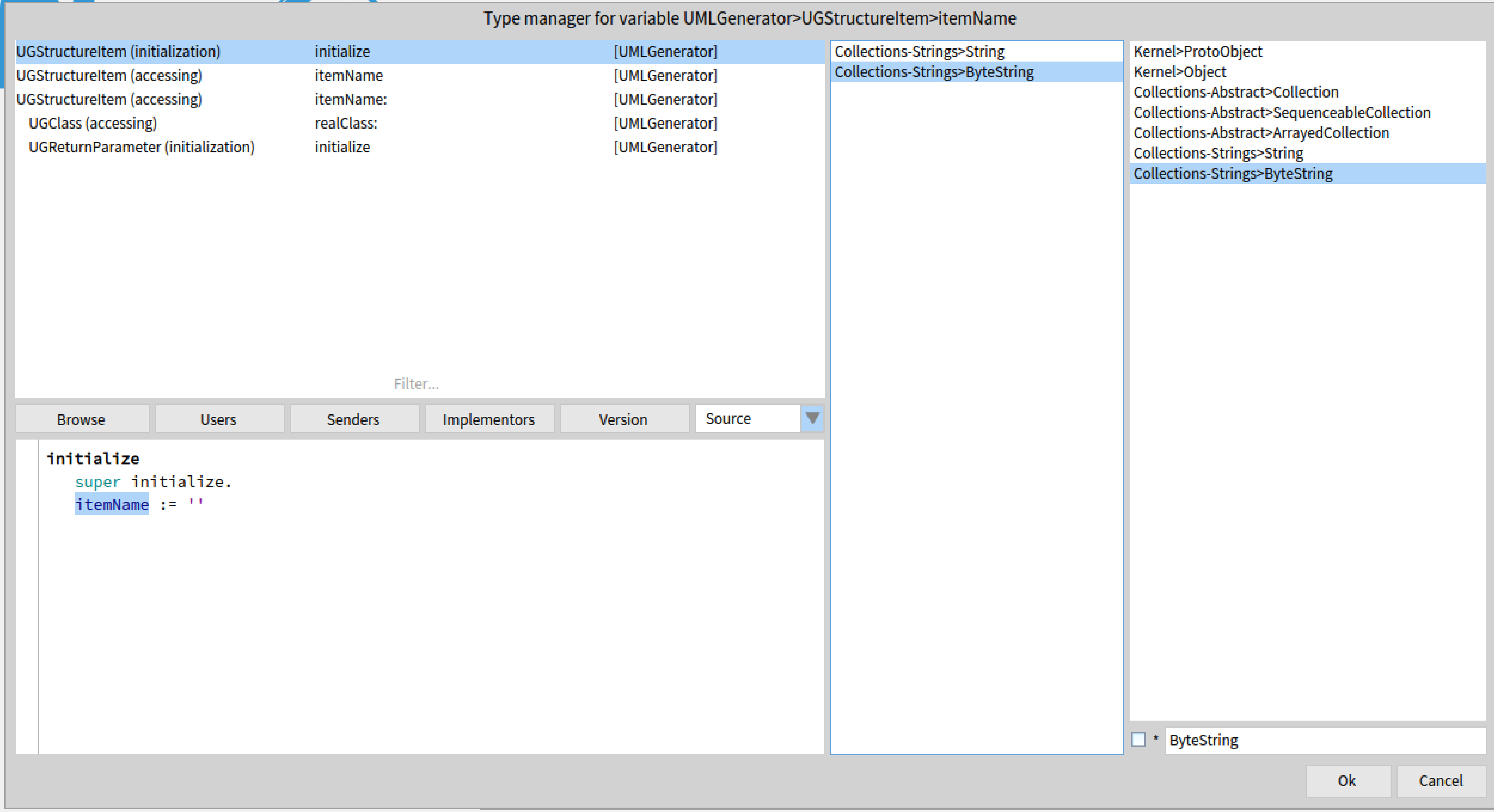
Picking one type from several options

- Semi-automatic way: provide user with as much insight as possible to make the decision
- Automatic way: Heuristics (prone to error) or picking common superclass

Semi-automatic way of picking a type

Help user by providing GUI with:

- Option to pick from classes offered by type inferrer(s),
- their superclasses
- or write any name of a class in the system
  - in case type inferrer did not find anything or offer wrong classes,
- integrated „references of the variable" GUI
  - currently just for instance variables, extendable



Type manager for variable UMLGenerator>UGStructureItem>itemName

UGStructureItem (initialization)	initialize	[UMLGenerator]
UGStructureItem (accessing)	itemName	[UMLGenerator]
UGStructureItem (accessing)	itemName:	[UMLGenerator]
UGClass (accessing)	realClass:	[UMLGenerator]
UGReturnParameter (initialization)	initialize	[UMLGenerator]

```
initialize
	super initialize.
	itemName := ''
```

Collections-Strings>String
Collections-Strings>ByteString

Kernel>ProtoObject
Kernel>Object
Collections-Abstract>Collection
Collections-Abstract>SequenceableCollection
Collections-Abstract>ArrayedCollection
Collections-Strings>String
Collections-Strings>ByteString

□ * ByteString

Automatic way of picking a type

Picking common superclass

- Discards correct type if there is also a wrong, unrelated type
- Duck typing – two classes without common superclass (except of Object) might both be correct
  - Example: a block might be subtituted by a symbol
- Prefers marking types as unknown/Object before picking the wrong one
- Sometimes class should be skipped to superclasses (SmallInteger -> Integer)

Smalltalk vs UML

- Representing String with multiplicity (0..*) and marking elements as unordered and unique
  - How to denote difference OrderedCollection and Array? What to do with Dictionaries?
  - String is also a collection – we prefer to represent it as String, not ordered collection of characters
- UML is based on Java-like languages
  - Static methods/variables, interfaces, no traits
  - What to do with class-side? Marking them as static or separating UML Class for String and another for String class?

Real-time type inference

- Custom implementation of real-time type inference
- Based on Metalinks – enhancing AST by custom code with types logging
  - before the AST root of the method itself => method arguments,
  - after each variable assignment node => contents of instance variables,
  - before each return node => returned types,
  - If last node is not return => returned type is self class (could be found statically).

Real-time type inference

- Cannot find types for methods not **executed** during logging & variables not assigned to
- Requires running application, well tested code, executable examples or at least very thorough use-case scenarios
  - otherwise nothing is found

Real-time type inferrer

- A command to start logging
- Manual execution of tests, examples…
- A command to stop logging and remove metalinks
- The inferrer is ready to by asked for logged types
  - Is provided to TypesManager that asks for types

Comparison of available type inferrers

- Took several packages in Pharo
- Used each type inferrer separatedly and all of them combined together
- Set automatic way of picking types
- In case of real-time type inference, executed all available tests and executable examples
- Counted items with a single found type other than Object
  - Items = instance variables, shared variables, method arguments, return types

Comparison of available type inferrers

Package name	Total	RoelTyper	RBRefactoryT.	Real-time	Combined
Zinc-HTTP	2369	76 (3,2 %)	88 (3,7 %)	1565 (66,1 %)	1584 (66,9 %)
Athens-Cairo	1146	34 (3,0 %)	31 (2,7 %)	298 (43,5 %)	513 (44,8 %)
GT-Playground	264	3 (1,1 %)	5 (1,9 %)	58 (22,0 %)	60 (22,7 %)
Ombu	430	21 (4,9 %)	26 (6,0 %)	255 (59,3 %)	263 (61,2 %)
OSWindow-Core	1675	57 (3,4 %)	50 (3,0 %)	64 (3,8 %)	125 (7,5 %)
Tool-Diff	559	34 (6,1 %)	29 (5,2 %)	0 (0,0 %)	34 (6,1 %)
Traits	1823	14 (0,8 %)	13 (0,7 %)	17 (0,9 %)	30 (1,6 %)

Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Class Diagrams with as many automation as possible?

How to transfer/import the generated model to Enterprise Architect?

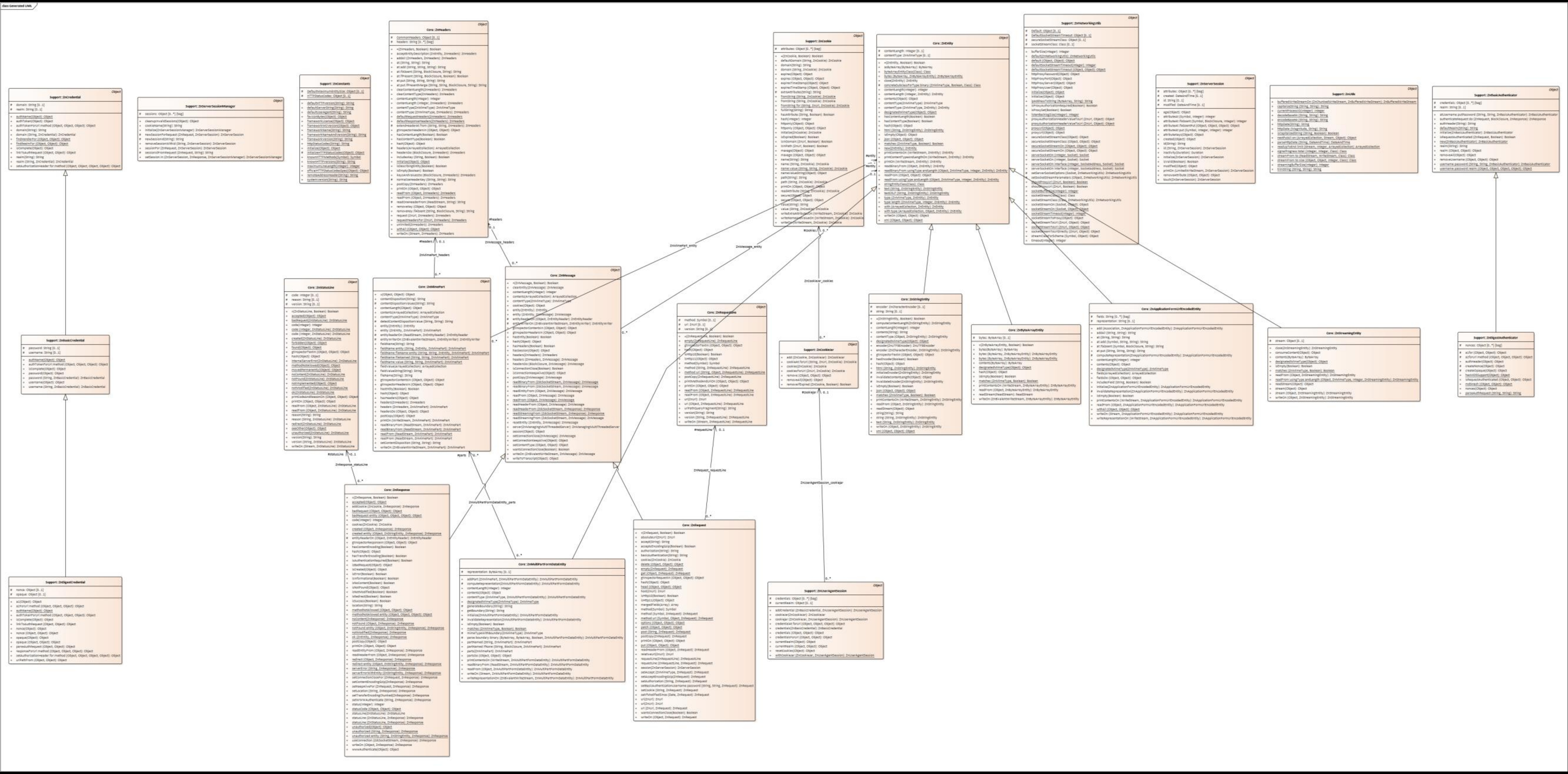
- Standardized XMI format for representing UML models
  - based on XML
- UML Metamodel for OpenPonk comes with XMI exporter
- Most modelling tools come with XMI import capabilities

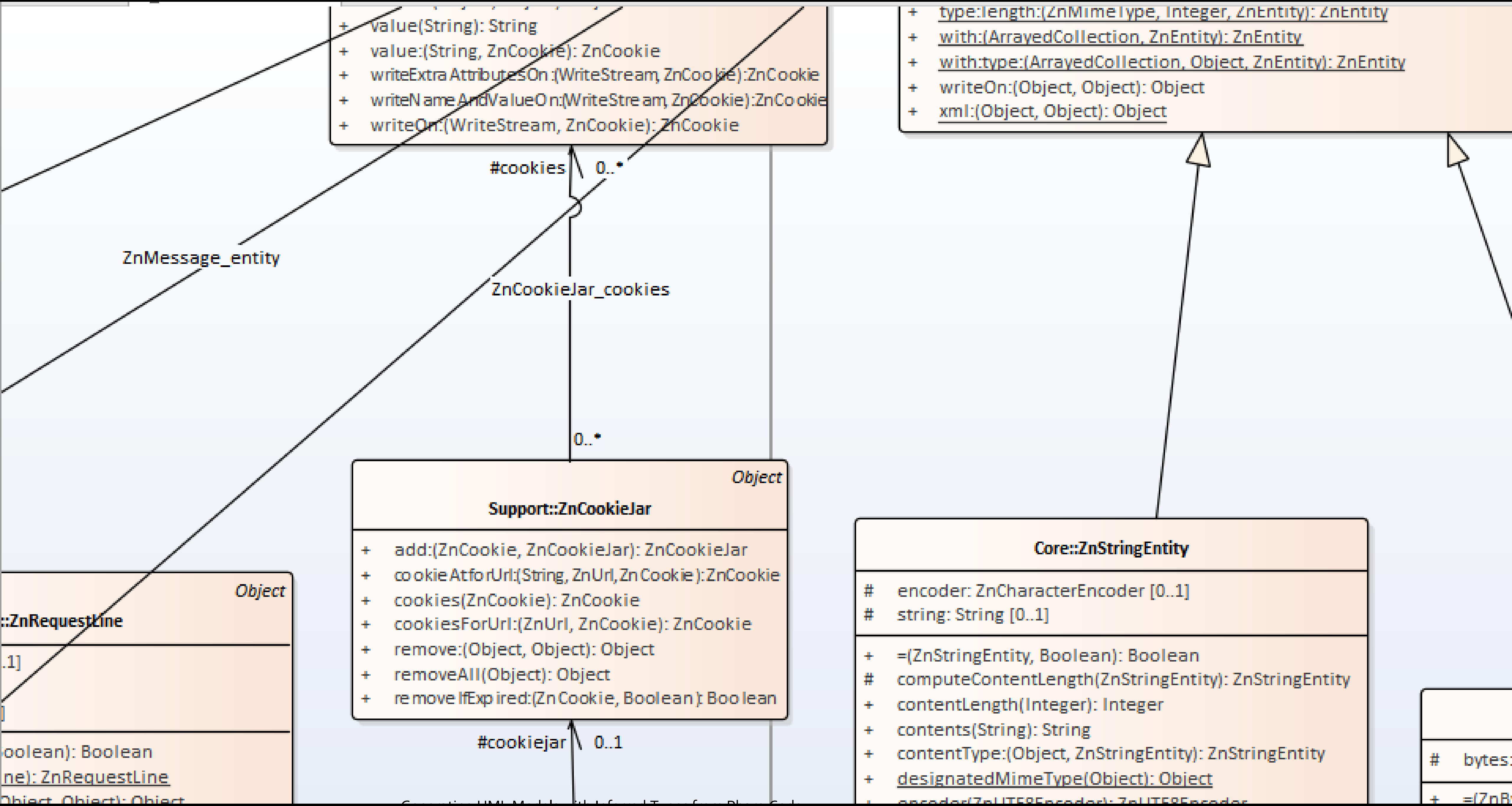
Steps towards the goal

- How to represent UML models in Pharo?
- How to generate a UML model with class structure, operations (methods), properties (variables) etc. from a Pharo code?
- How to find types of instance variables, method parameters and return types?
- How to transfer/import the generated model to Enterprise Architect?
- How could be such model used for creation of UML Class Diagrams with as many automation as possible?

How could be such model used for creation of UML Class Diagrams with as many automation as possible?

- TODO ☺
- We can generate one large diagram of the whole package or tag
  - Might be too large with unimportant auxiliary classes
- How to split the model into several class diagrams?
- For now, Enterprise Architect offers quite quick way to select all classes in any package/tag and *(quite good)* auto-layouting


class Generated UML
Support::ZnCredential
Support::ZnServerSessionManager
Support::ZnConstants
Core::ZnHeaders
Support::ZnCookie
Core::ZnEntity
Support::ZnNetworkingUtils
Support::ZnServerSession
Support::ZnUtils
Support::ZnBasicAuthenticator
Core::ZnStatusLine
Core::ZnMimePart
Core::ZnMessage
Core::ZnRequestLine
Support::ZnCookieJar
Core::ZnStringEntity
Core::ZnByteArrayEntity
Core::ZnApplicationFormUrlEncodedEntity
Core::ZnStreamingEntity
Support::ZnDigestAuthenticator
Support::ZnBasicCredential
Support::ZnDigestCredential
Core::ZnResponse
Core::ZnMultiPartFormDataEntity
Core::ZnRequest
Support::ZnUserAgentSession
ZnMimePart_headers
ZnMessage_headers
ZnMimePart_entity
ZnMessage_entity
ZnCookieJar_cookies
ZnResponse_statusLine
ZnMultiPartFormDataEntity_parts
ZnRequest_requestLine
ZnUserAgentSession_cookieJar
Object



+ value(String): String
+ value:(String, ZnCookie): ZnCookie
+ writeExtraAttributesOn:(WriteStream, ZnCookie):ZnCookie
+ writeNameAndValueOn:(WriteStream, ZnCookie):ZnCookie
+ writeOn:(WriteStream, ZnCookie): ZnCookie

+ type:length:(ZnMimeType, Integer, ZnEntity): ZnEntity
+ with:(ArrayedCollection, ZnEntity): ZnEntity
+ with:type:(ArrayedCollection, Object, ZnEntity): ZnEntity
+ writeOn:(Object, Object): Object
+ xml:(Object, Object): Object

Object

Support::ZnCookieJar

+ add:(ZnCookie, ZnCookieJar): ZnCookieJar
+ cookieAtforUrl:(String, ZnUrl, ZnCookie):ZnCookie
+ cookies(ZnCookie): ZnCookie
+ cookiesForUrl:(ZnUrl, ZnCookie): ZnCookie
+ remove:(Object, Object): Object
+ removeAll(Object): Object
+ removeIfExpired:(ZnCookie, Boolean): Boolean

Object

::ZnRequestLine

.1]

]

oolean): Boolean
ne): ZnRequestLine
Object, Object): Object

Core::ZnStringEntity

encoder: ZnCharacterEncoder [0..1]
string: String [0..1]

+ =(ZnStringEntity, Boolean): Boolean
# computeContentLength(ZnStringEntity): ZnStringEntity
+ contentLength(Integer): Integer
+ contents(String): String
+ contentType:(Object, ZnStringEntity): ZnStringEntity
+ designatedMimeType(Object): Object

bytes:
+ =(ZnB

Future work

- Generating diagrams from models
- Generating models for sequence diagrams
- Improving current type inference tools
- Improving several important aspects of current solution
- Trying the solution on large commercial project

Thank you

Questions?

Jan Blizničenko and Robert Pergl

Czech Technical University in Prague

The presented research was sponsored by Tomcat® computer GmbH.