CommIT
Computer Science & Information Technologies

Universidad Rey Juan Carlos

SmaCQA: from Business Models to Smart Contracts

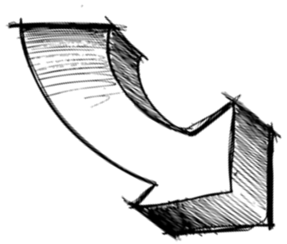
URJC Escuela Técnica Superior de Ingeniería Informática

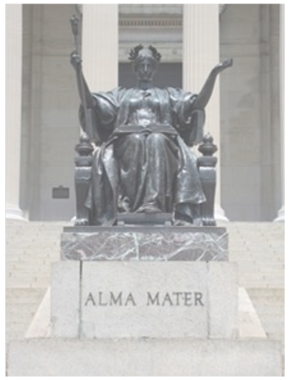
Juan Manuel Vara [Juancho]
juanmanuel.vara@urjc.es
@jmvara

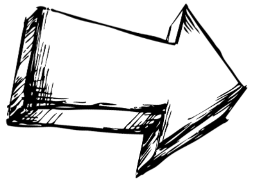
The speaker

The beginning

- Ingeniería Aeronáutica - UCM
- Ingeniería Técnica en Telecomunicaciones - UPM
- Ciencias Ambientales - URJC
- Ingeniería Informática – URJC
- IBERIA – Viva Tours







The speaker

My life at @URJC

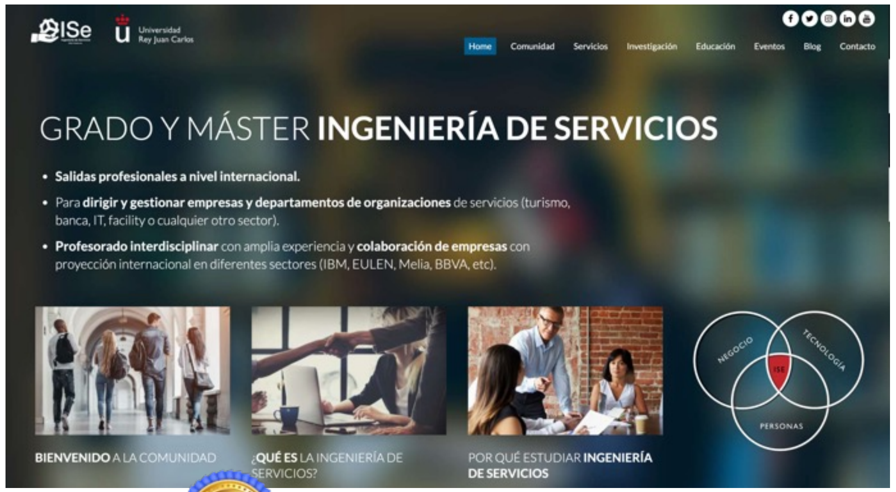
- Ingeniería Informática - 2004
- Master en Tecnologías de la Información y Sistemas Informáticos – 2005
- Diploma de Estudios Avanzados - 2006
- Doctorado en Informática y Modelización Matemática – 2009

- Profesor Ayudante – 2005 – 2009
- Profesor Ayudante Doctor – 2009 – 2010
- Profesor Titular de Universidad Interino – 2010 - 2014
- Profesor Contratado Doctor – 2014 - 2018
- Profesor Titular de Universidad – 2018 - 2023
- Catedrático de Universidad – 2023 …

The speaker

My life at @URJC – Management Duties

- Head of the BsC in Services Engineering
- Head of the MsC in Information Systems Engineering
- Head of the SE, Informations Systems and Service Engineering at the PhD Program on TIC



GRADO Y MÁSTER **INGENIERÍA DE SERVICIOS**

- **Salidas profesionales a nivel internacional.**
- Para **dirigir y gestionar empresas y departamentos de organizaciones** de servicios (turismo, banca, IT, facility o cualquier otro sector).
- **Profesorado interdisciplinar** con amplia experiencia y **colaboración de empresas** con proyección internacional en diferentes sectores (IBM, EULEN, Melia, BBVA, etc).

BIENVENIDO A LA COMUNIDAD

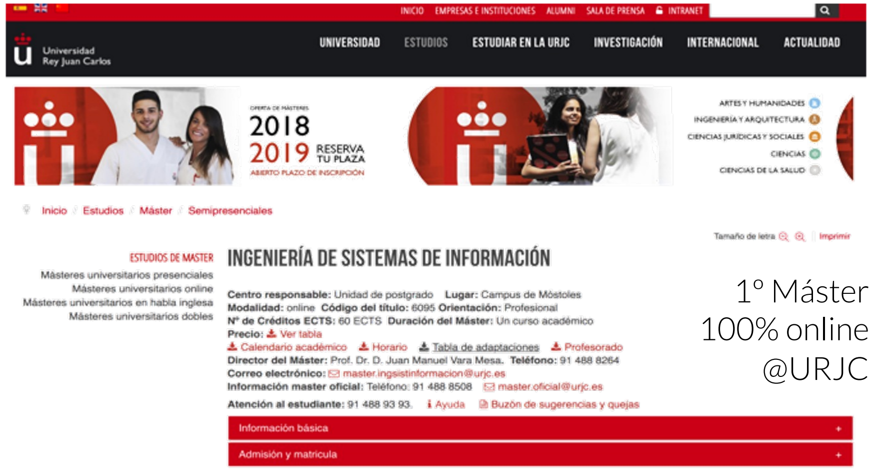
¿QUÉ ES LA INGENIERÍA DE SERVICIOS?

POR QUÉ ESTUDIAR **INGENIERÍA DE SERVICIOS**

ISSIP Excellence In Service Innovation Award

IBM Academic Awards
Supporting research and curriculum innovation.
IBM Global University Programs

The International Society of **Service Innovation** Professionals



Inicio / Estudios / Máster / Semipresenciales

Tamaño de letra | Imprimir

ESTUDIOS DE MASTER
Másteres universitarios presenciales
Másteres universitarios online
Másteres universitarios en habla inglesa
Másteres universitarios dobles

INGENIERÍA DE SISTEMAS DE INFORMACIÓN

Centro responsable: Unidad de postgrado **Lugar:** Campus de Móstoles
Modalidad: online **Código del título:** 6095 **Orientación:** Profesional
Nº de Créditos ECTS: 60 ECTS **Duración del Máster:** Un curso académico
Precio: Ver tabla
Calendario académico Horario Tabla de adaptaciones Profesorado
Director del Máster: Prof. Dr. D. Juan Manuel Vara Mesa. **Teléfono:** 91 488 8264
Correo electrónico: master.ingsistinformacion@urjc.es
Información master oficial: Teléfono: 91 488 8508 master.oficial@urjc.es
Atención al estudiante: 91 488 93 93. Ayuda Buzón de sugerencias y quejas

Información básica +
Admisión y matrícula +

1º Máster
100% online
@URJC

PROGRAMA DE DOCTORADO EN TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES

Atención al estudiante: Buzón de Ayuda al Estudiante Buzón de sugerencias y quejas
Más información: Escuela Internacional de Doctorado

Coordinador y comisión académica +
Líneas de investigación

[illegible]

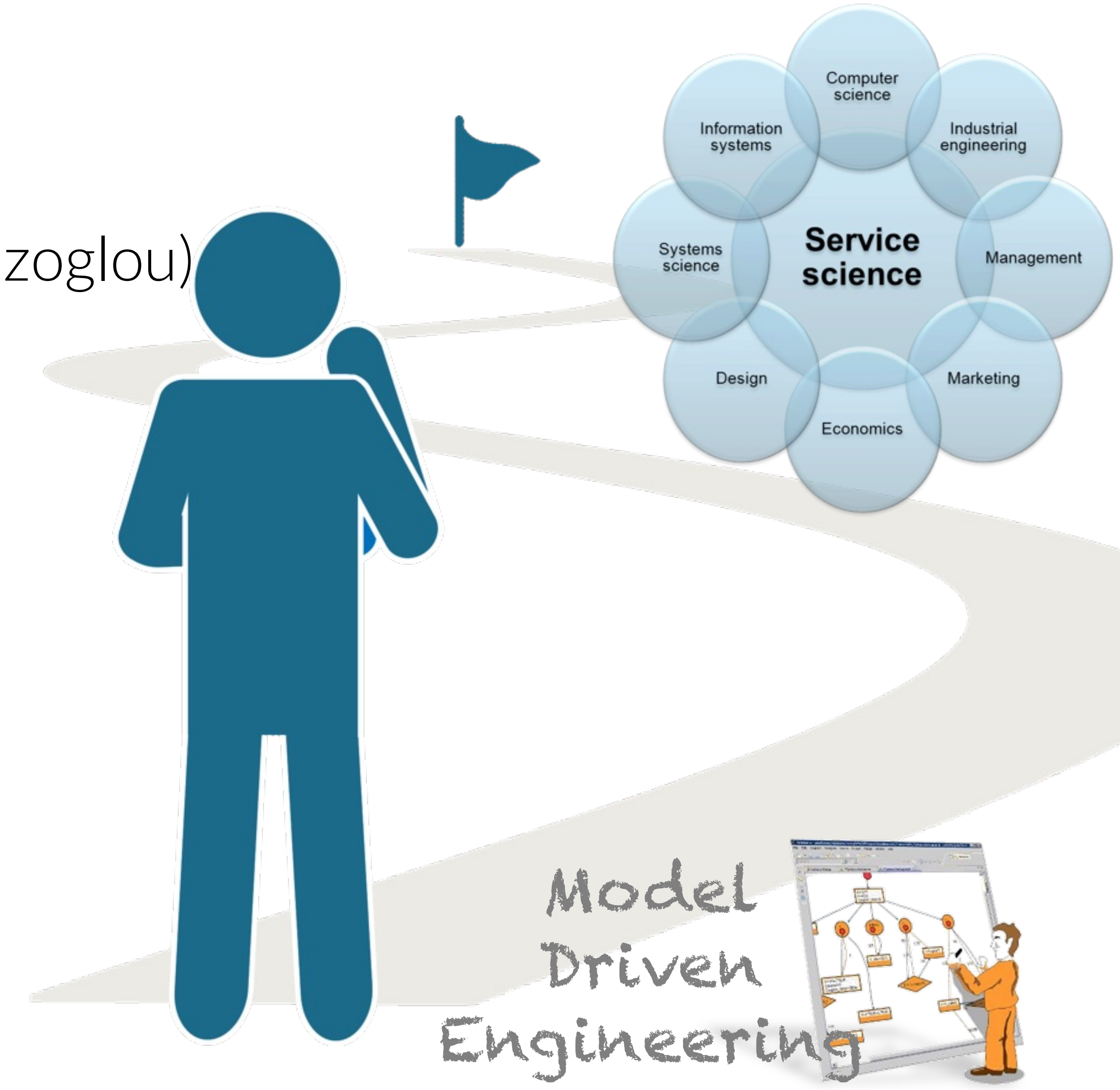
Ingeniería del software, sistemas de información y servicios

[illegible]

The speaker

My life as a researcher

- Pre-doc at U. Nantes (Jean Bézivin)
- Post-doc at Tilburg University (Mike Papazoglou)
- 2 Sexenios / 1 Transferencia
- 5 Docentia
- 7 Tesis Doctorales en los últimos 10 años
- 24 artículos JCR – 25 congresos CORE
- IP proyectos MINECO
- Investigador H2020
- …



The speaker

From MDE Tooling …

M2DAT
Technical and methodlogical decisions to build MDE tools

MeTAGeM
Model-driven development of model transformations

MeTAGeM-Trace
Model-driven development of model transformations supporting traces generation

iTrace
Management and analysis of traceability information in MDE projects

CEViNEdit
Enabling Cognitive Effectiveness in the Visual Notation of Graphic Editors

… to MDE-based tooling

M2DAT-DB

kyBDele

SRMod

INNoVaServ Toolkit …A New Way to Design Services…

Agenda

Blockchain

THE BIGGEST INVENTION SINCE THE EMERGENCE OF THE INTERNET!!
https://www.healthnewsreview.org/wp-content/uploads/2016/12/iStock-487078483.jpg

Blockchain

What is a Blockchain?

- A distributed DB + Encryption + Immutability + stored procedures (smart contracts)
  - A list (chain) of groups (blocks) of transactions
  - Like traditional DDBBs, they can be used for anything a DB is used.

How does it work?

- Interested subjects add transactions to the pool
- Nodes verify and add them to some block on the ledger
- Ledger is replicated among distributed nodes
- Eventual consistency
  - Absence of centralized control: all nodes achieve consensus about the ledger's content
- Append-only data structure
  - May add transactions – Nearly imposible to change data

Blockchain

Why so much hype?

- Digitalization
  - Goods and services become inmaterial
  - Music ⇒ Inmaterial nature + low costs of data transfer
  - Benefit from the advantages of p2p systems

Disintermediation

- A tool for **achieving and maintaining integrity** in purely distributed p2p systems … … that consists of an unknown number of peers with unknown reliability and trustworthiness

Ownership and Witnesses

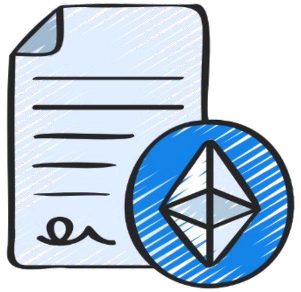
- Having one witness is good, but having many independent witnesses is the key.
- Instead of one ledger, a p2p system of ledgers …

Smart Contracts

Computer programs

Szabo, N. (1996). Smart contracts: building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought,(16)*, 18, 2.

- Executes **autonomously** the clauses collected in it when the conditions are satisfied
  - DTL as a DDBB
  - Smart (*Submissive*) Contracts as triggers or microservices where the business logic transacting with that data lives
- Blockchain technology "Sets in stone" the agreement



SOLIDITY

1. Conditions are programmed
2. Implied parties sign the conditions (program)
3. Contract is *placed* into a blockchain so no one could modify it

➩

- Programs does not leave space to different interpretations ➩ Disambiguation
- No need of a trusted third-party ➩ Disintermediation | ↯Transaction Costs
- Data Storage (future disputes) ➩ Inmutability
- Public ➩ Transparency
- Automatic execution ➩ Time-saving

VS Conventional Contracts

Smart Contracts

Ecosystem

Blockchain

TRON

CARDANO

AVALANCHE

ethereum

Language

vyper

JS

Go

Rust

python

clarity

CAIRO

SOLIDITY

```
Libraries & Interfaces

Global variables
```

```
Events / Modifiers

Contract signature (is)

Constructor
```

```
Functions
```

Similar to a Class in any OOPL

Dealing with Smart Contracts | Issues

Learning Curve

- Alharby, M., Aldweesh, A., & van Moorsel, A. (2018). Blockchain-based smart contracts: A systematic mapping study of academic research (2018). In *Proceedings of the 2018 International Conference on Cloud Computing, Big Data and Blockchain*.

Security Issues

- Mavridou, A., & Laszka, A. (2018, February). Designing secure ethereum smart contracts: A finite state machine based approach. In *International Conference on Financial Cryptography and Data Security* (pp. 523-540). Springer, Berlin, Heidelberg.

IT – Business Gap

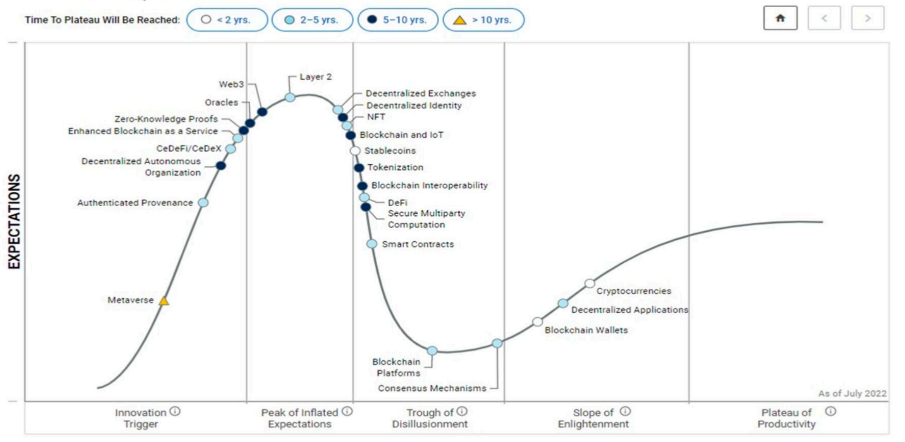
- Mik, E. (2017). Smart contracts: terminology, technical limitations and real world complexity. *Law, Innovation and Technology, 9*(2), 269-300.
- Bosu, A., Iqbal, A., Shahriyar, R., & Chakraborty, P. (2019). Understanding the motivations, challenges and needs of blockchain software developers: A survey. *Empirical Software Engineering, 24*(4), 2636-2673.

"In other words, they're code that does what it's been programmed to do.

If the **business rules** ... have been defined badly and/or the programmer doesn't do a good job, the result is going to be a mess, and, even if programmed correctly, a smart contract isn't smart – it just functions as **designed**."

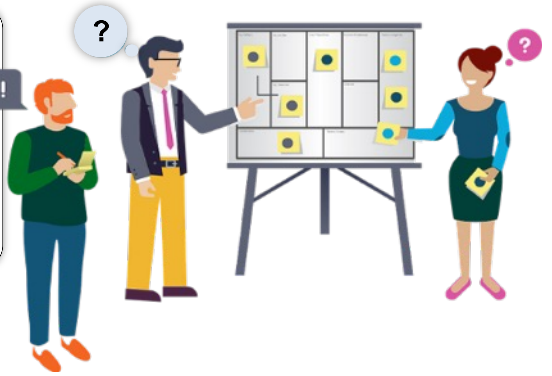
What's a smart contract (and how does it work)?
Computer World, Jul 29 (2019)

Gartner Hype Cycle for Blockchain and Web3, 2022



Dealing with Smart Contracts | Issues

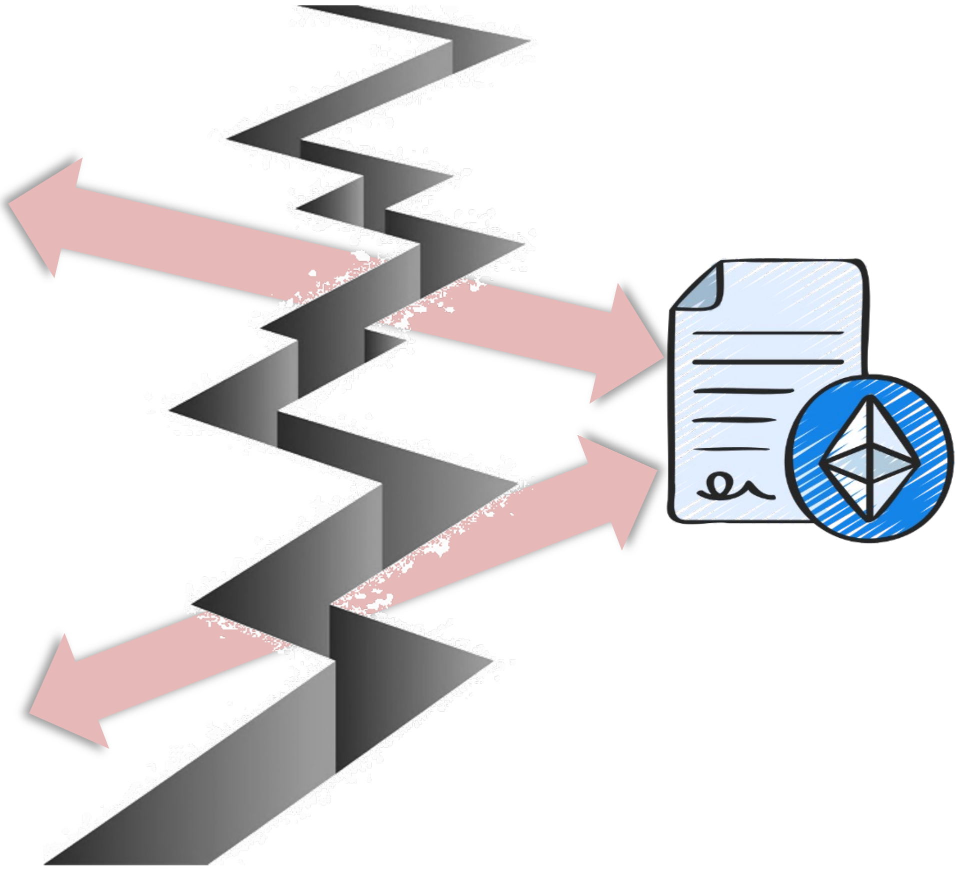
Hard to understand and deal with for business professionals



Business Professionals

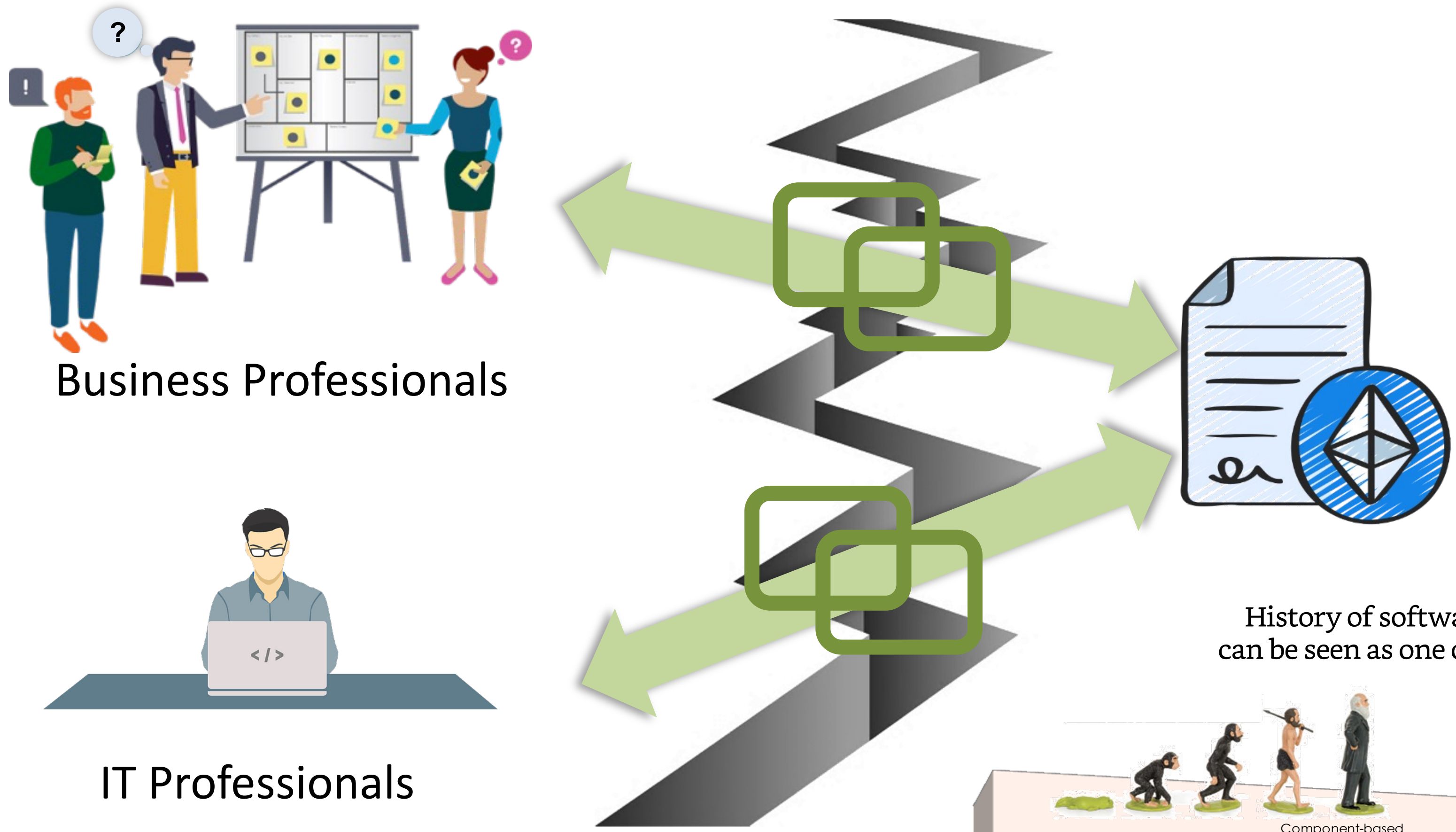
Essential + Accidental Complexity
for Developers

IT Professionals

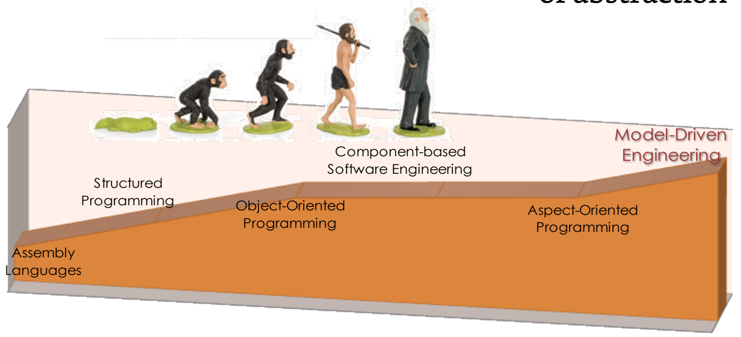


Dealing with Smart Contracts | Issues

Democratize Smart Contract creation & management



History of software engineering can be seen as one of raising levels of abstraction


Model-Driven Engineering
Component-based Software Engineering
Structured Programming
Object-Oriented Programming
Aspect-Oriented Programming
Assembly Languages

Taking the most from Models

Simplify Smart Contract coding

- Auto-completion
- Syntactical validation
- QuickFixes
- Auto-documentation

Promote reliability and security

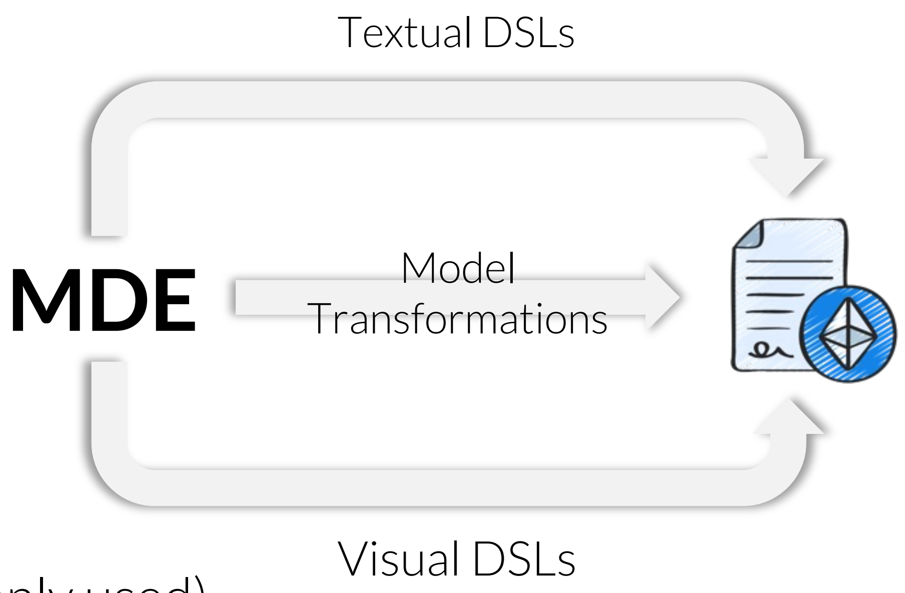
- Generating template code (analyzed and tested code is only used)
- Good practices & risk patterns

Simplify Smart Contract development

- Visual editors

Close the gap between business professionals and developers

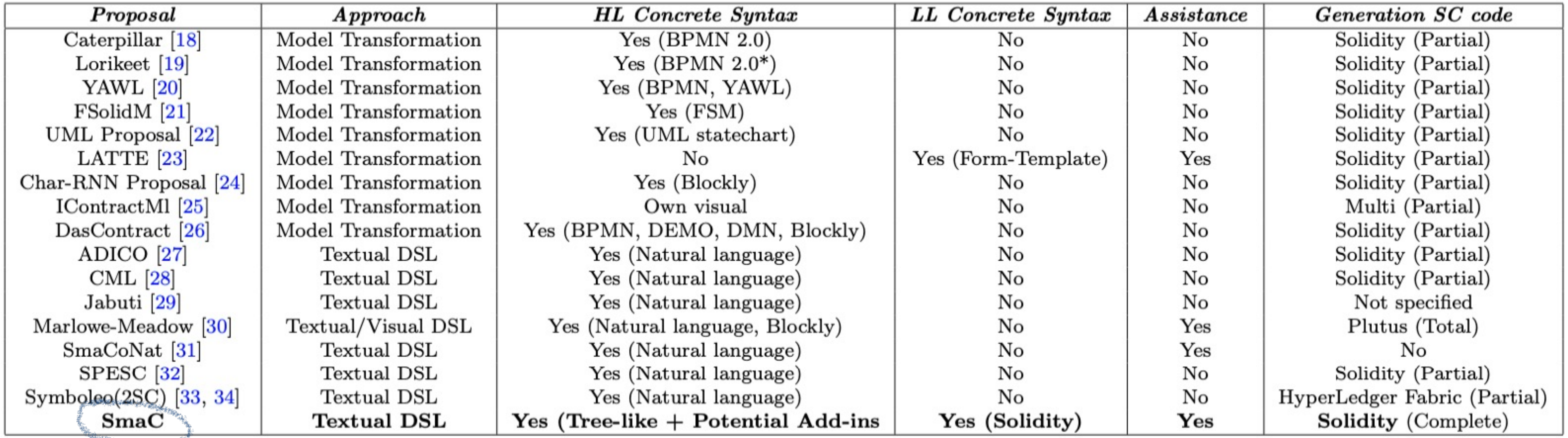
- Development of technological bridges



Model-driven engineering uses models as the primary source of information, abstracting the complexity of the system facilitating its specification, design, implementation, and verification.

Dealing with Smart Contract issues

Model-based proposals to deal with Smart Contracts



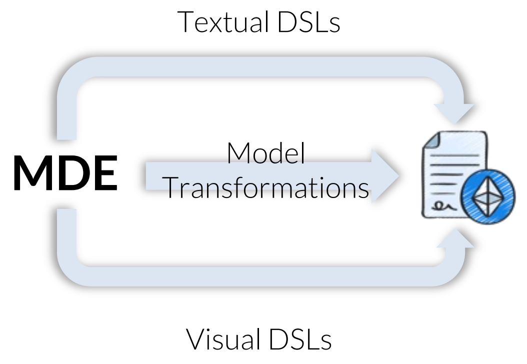
Proposal	*Approach*	*HL Concrete Syntax*	*LL Concrete Syntax*	*Assistance*	*Generation SC code*
Caterpillar [18]	Model Transformation	Yes (BPMN 2.0)	No	No	Solidity (Partial)
Lorikeet [19]	Model Transformation	Yes (BPMN 2.0*)	No	No	Solidity (Partial)
YAWL [20]	Model Transformation	Yes (BPMN, YAWL)	No	No	Solidity (Partial)
FSolidM [21]	Model Transformation	Yes (FSM)	No	No	Solidity (Partial)
UML Proposal [22]	Model Transformation	Yes (UML statechart)	No	No	Solidity (Partial)
LATTE [23]	Model Transformation	No	Yes (Form-Template)	Yes	Solidity (Partial)
Char-RNN Proposal [24]	Model Transformation	Yes (Blockly)	No	No	Solidity (Partial)
IContractMl [25]	Model Transformation	Own visual	No	No	Multi (Partial)
DasContract [26]	Model Transformation	Yes (BPMN, DEMO, DMN, Blockly)	No	No	Solidity (Partial)
ADICO [27]	Textual DSL	Yes (Natural language)	No	No	Solidity (Partial)
CML [28]	Textual DSL	Yes (Natural language)	No	No	Solidity (Partial)
Jabuti [29]	Textual DSL	Yes (Natural language)	No	No	Not specified
Marlowe-Meadow [30]	Textual/Visual DSL	Yes (Natural language, Blockly)	No	Yes	Plutus (Total)
SmaCoNat [31]	Textual DSL	Yes (Natural language)	No	Yes	No
SPESC [32]	Textual DSL	Yes (Natural language)	No	No	Solidity (Partial)
Symboleo(2SC) [33, 34]	Textual DSL	Yes (Natural language)	No	No	HyperLedger Fabric (Partial)
SmaC	**Textual DSL**	**Yes (Tree-like + Potential Add-ins**	**Yes (Solidity)**	**Yes**	**Solidity** (Complete)

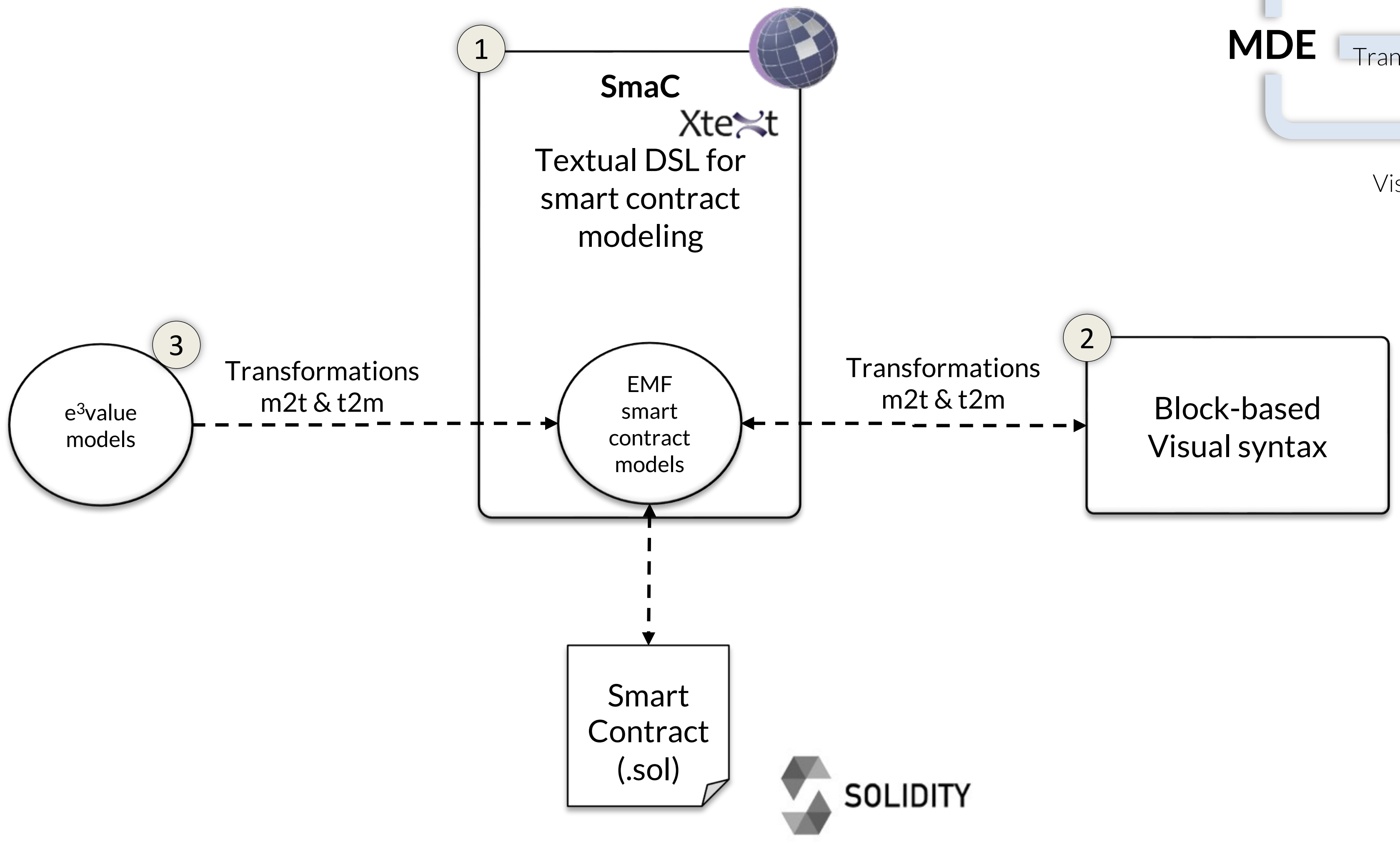
Integrating Smart Contracts into the Modeling Paradigm to Harness the Potential of Models. Gómez-Macías, D., Pérez-Blanco, F.J., Granada, D. Vara, J.M. *Software & Systems Modeling*, 2024 (Accepted for publication)

Agenda

Agenda

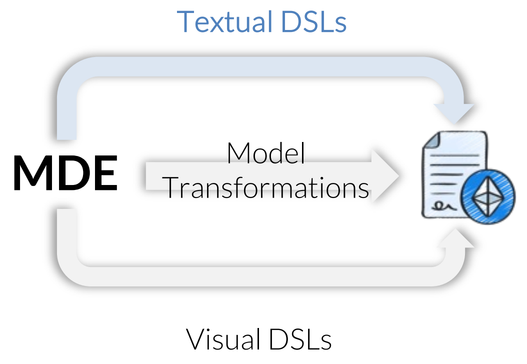
Research Proposal (1.0)

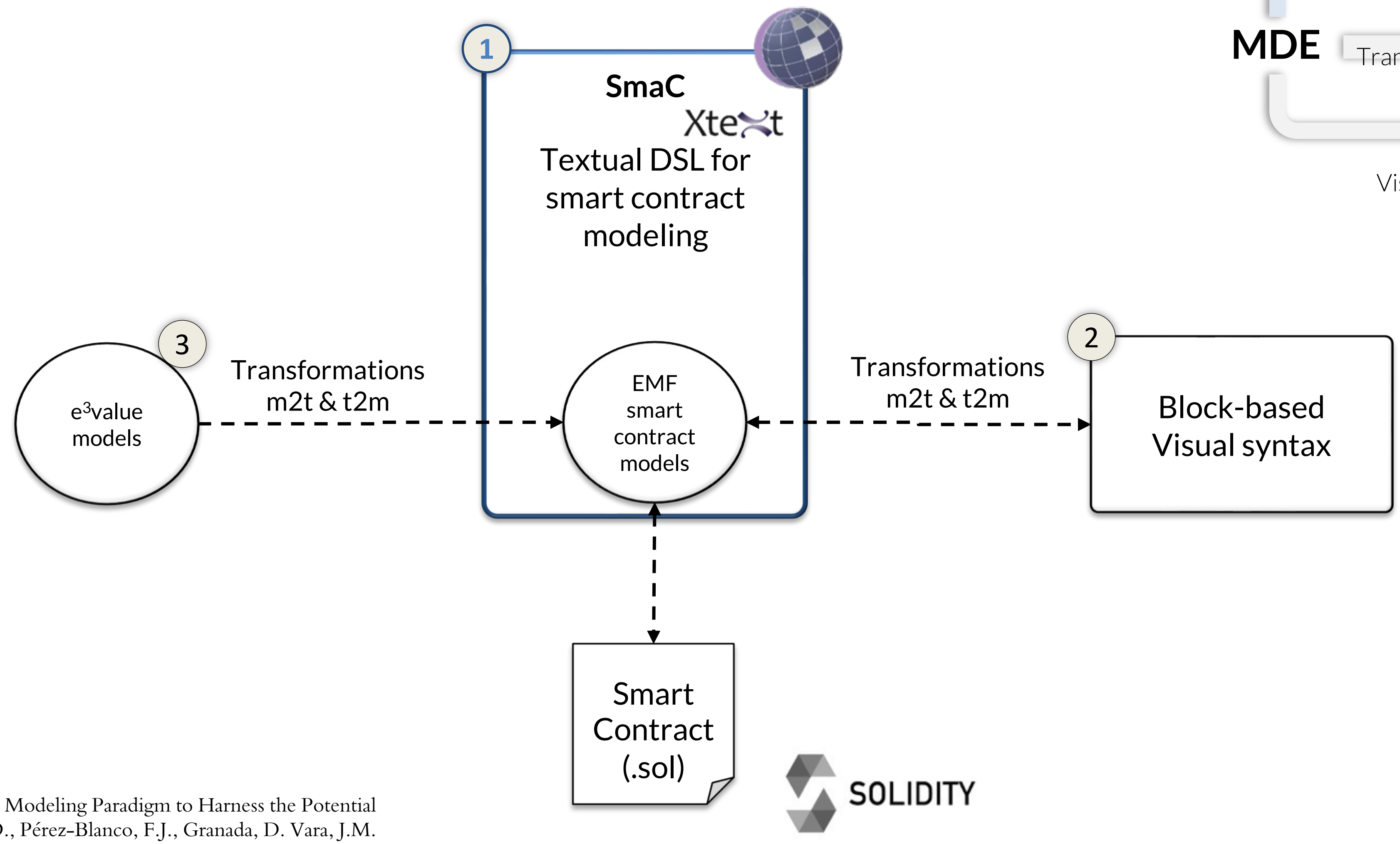




1
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
3
e^{3}value models
Transformations m2t & t2m
Transformations m2t & t2m
2
Block-based Visual syntax
Smart Contract (.sol)
SOLIDITY

Research Proposal (1.0) - SmaC



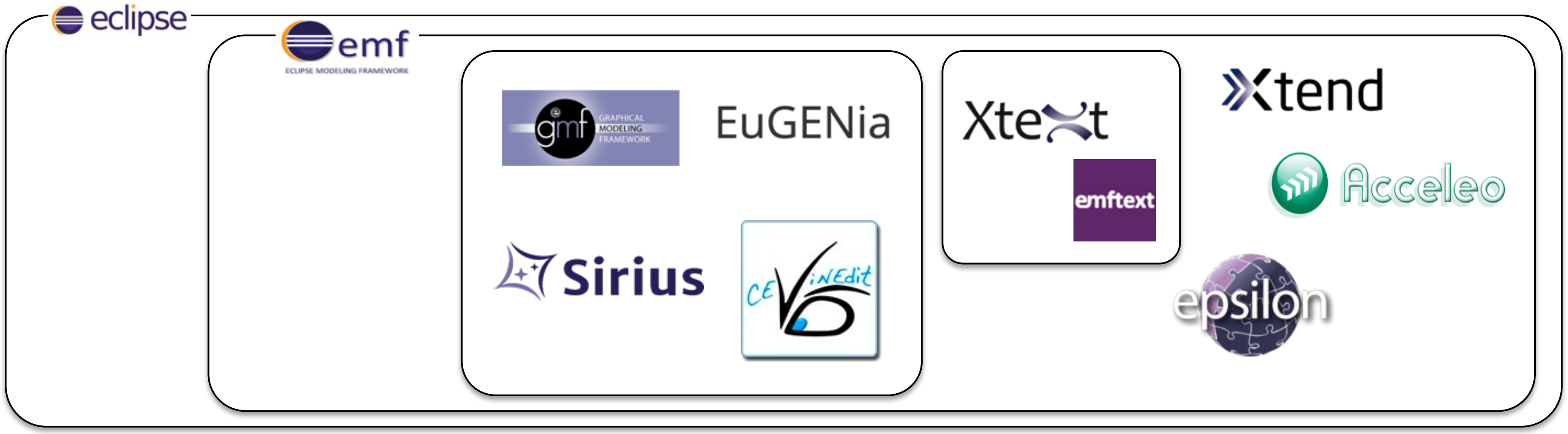

1
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
3
e³value models
Transformations m2t & t2m
2
Transformations m2t & t2m
Block-based Visual syntax
Smart Contract (.sol)
SOLIDITY

Integrating Smart Contracts into the Modeling Paradigm to Harness the Potential of Models. Gómez-Macías, D., Pérez-Blanco, F.J., Granada, D. Vara, J.M. *Software & Systems Modeling*, 2024 (Accepted for publication)

Xtext

What is Xtext?

- Framework for textual DSLs development
- Xtend (Java-like) for the development of validations, quickfixes, etc.
- Ecore metamodel automatically generated from the grammar.

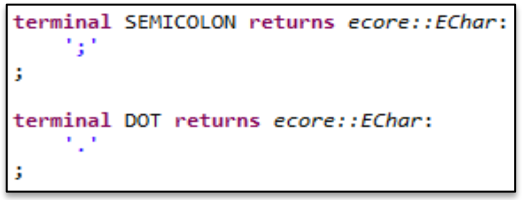


Develop a textual language using Xtext(I)

How to develop a textual language?

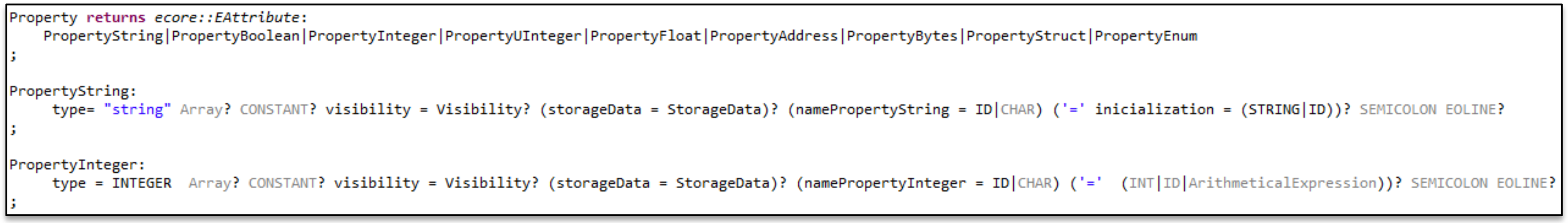
1. Write the grammar

   a) Define the terminals.

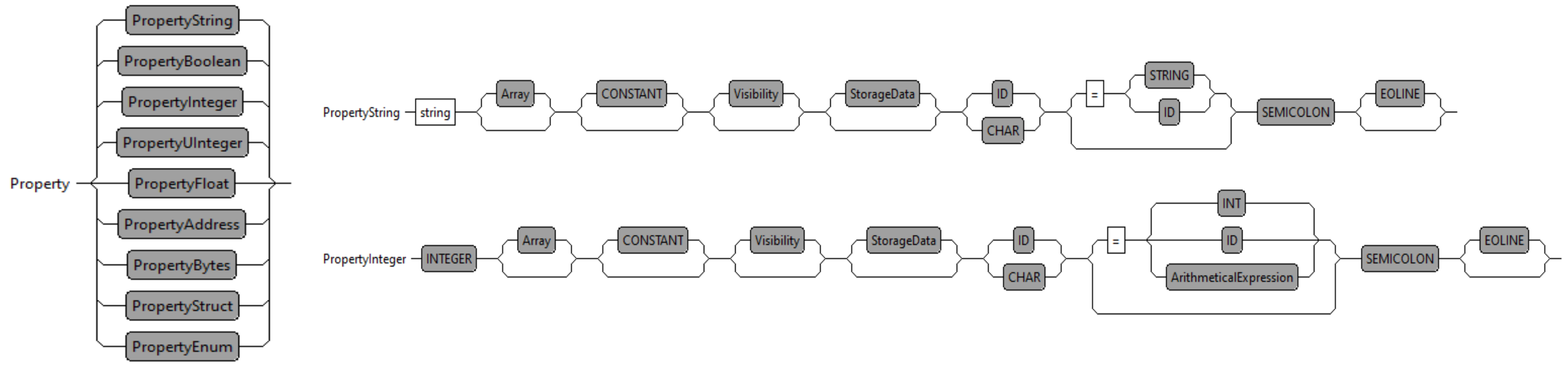


```
terminal SEMICOLON returns ecore::EChar:
    ';'
;

terminal DOT returns ecore::EChar:
    '.'
;
```

   b) Define the rules.



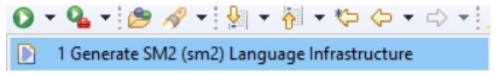
```
Property returns ecore::EAttribute:
    PropertyString|PropertyBoolean|PropertyInteger|PropertyUInteger|PropertyFloat|PropertyAddress|PropertyBytes|PropertyStruct|PropertyEnum
;

PropertyString:
    type= "string" Array? CONSTANT? visibility = Visibility? (storageData = StorageData)? (namePropertyString = ID|CHAR) ('=' inicialization = (STRING|ID))? SEMICOLON EOLINE?
;

PropertyInteger:
    type = INTEGER  Array? CONSTANT? visibility = Visibility? (storageData = StorageData)? (namePropertyInteger = ID|CHAR) ('='  (INT|ID|ArithmeticalExpression))? SEMICOLON EOLINE?
;
```



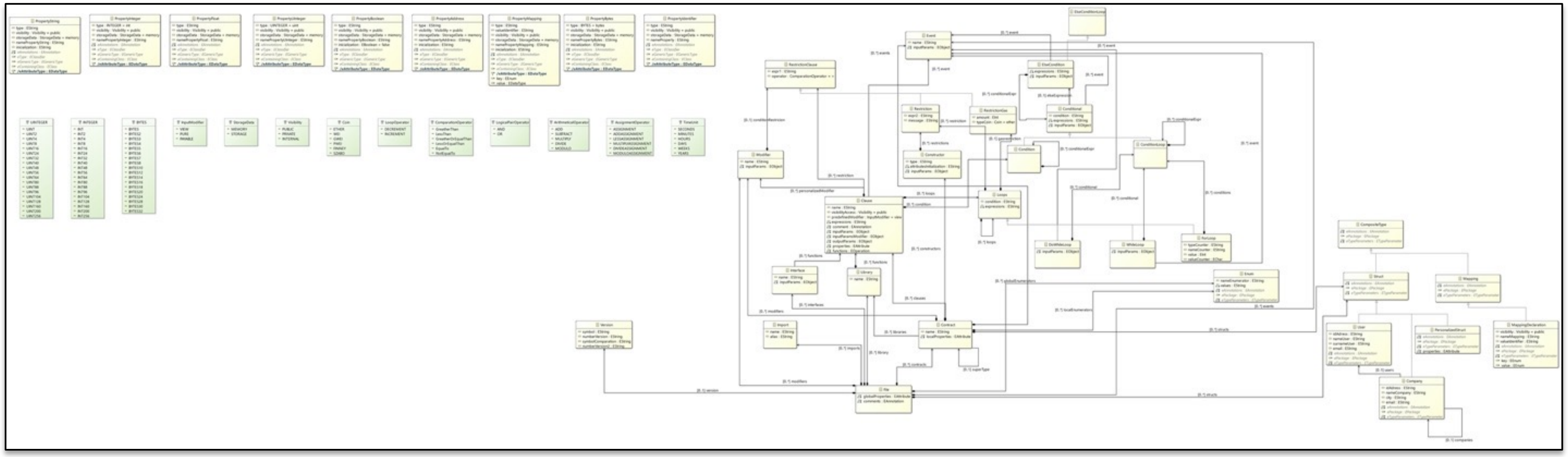
Develop a textual language using Xtext(II)

How to develop a textual language?

3. Generate language artifacts.



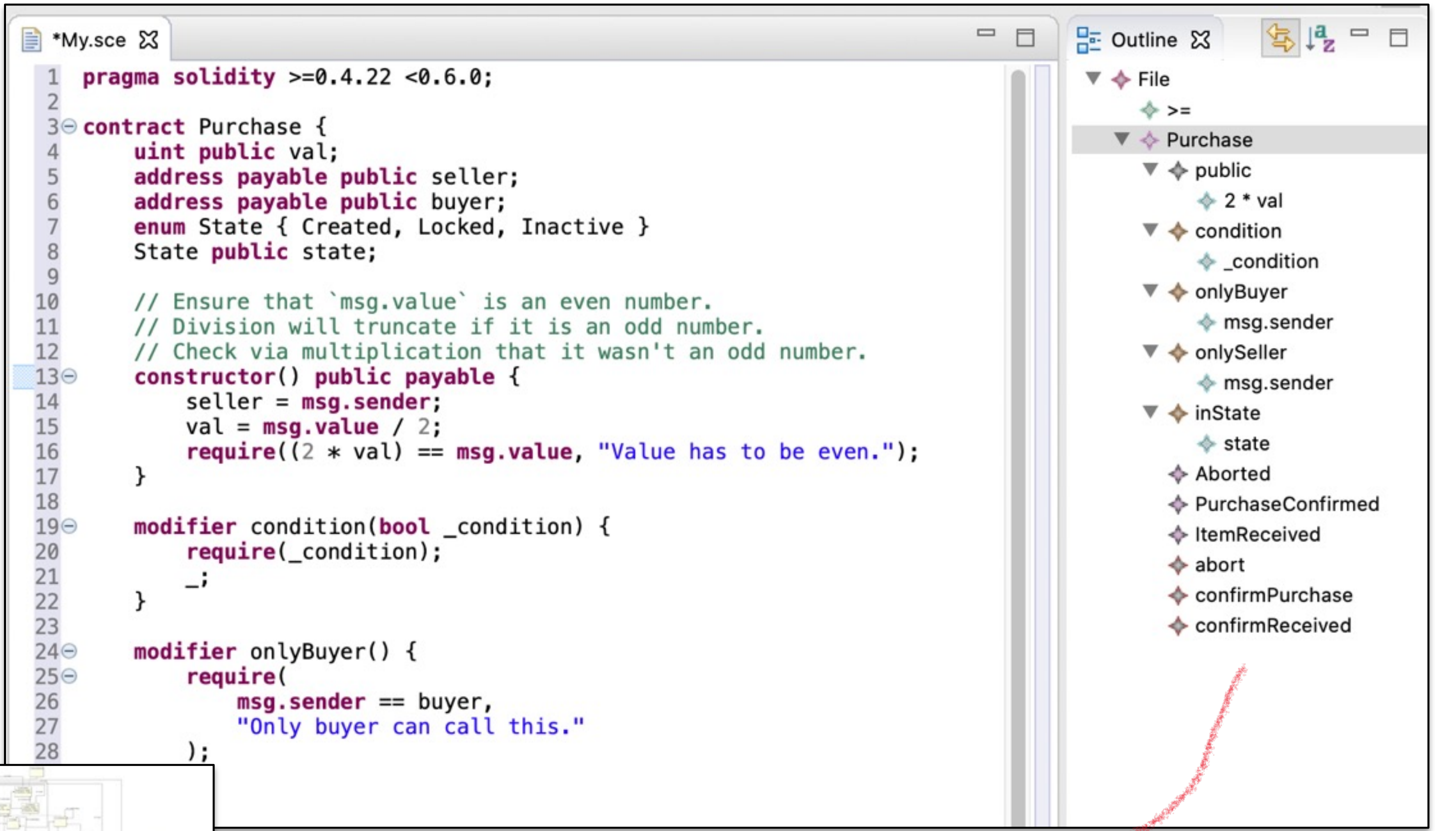
1 Generate SM2 (sm2) Language Infrastructure



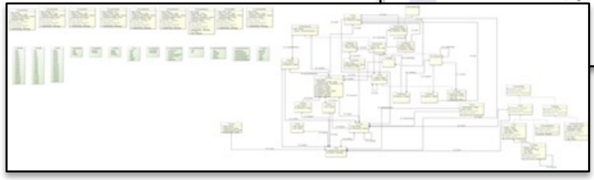
Develop a textual language using Xtext(II)

How to develop a textual language?

4. Run the Generated Eclipse plug-in.



```
pragma solidity >=0.4.22 <0.6.0;

contract Purchase {
    uint public val;
    address payable public seller;
    address payable public buyer;
    enum State { Created, Locked, Inactive }
    State public state;

    // Ensure that `msg.value` is an even number.
    // Division will truncate if it is an odd number.
    // Check via multiplication that it wasn't an odd number.
    constructor() public payable {
        seller = msg.sender;
        val = msg.value / 2;
        require((2 * val) == msg.value, "Value has to be even.");
    }

    modifier condition(bool _condition) {
        require(_condition);
        _;
    }

    modifier onlyBuyer() {
        require(
            msg.sender == buyer,
            "Only buyer can call this."
        );
```



Conforms To

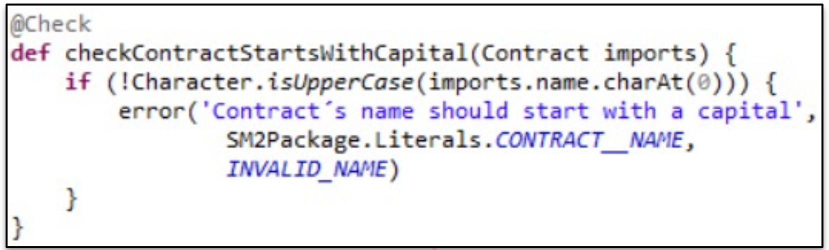
Develop a textual language using Xtext(II)

How to develop a textual language?

5. [Generate Code Generator - Xtend]

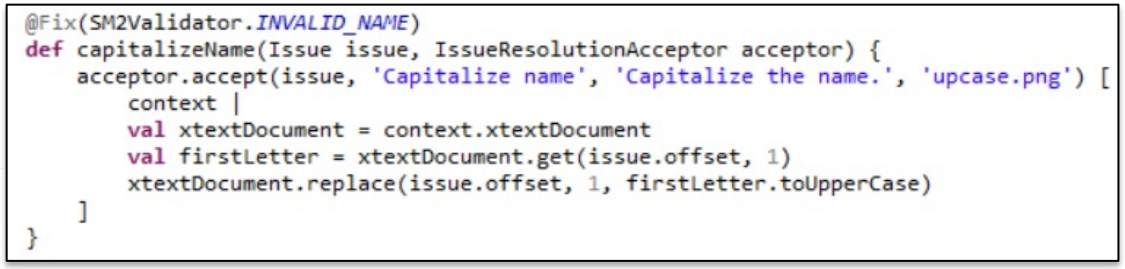
6. [Unit Testing]

7. [Creating Custom Validation Rules]



```
@Check
def checkContractStartsWithCapital(Contract imports) {
    if (!Character.isUpperCase(imports.name.charAt(0))) {
        error('Contract´s name should start with a capital',
                SM2Package.Literals.CONTRACT__NAME,
                INVALID_NAME)
    }
}
```

Including quickfixes



```
@Fix(SM2Validator.INVALID_NAME)
def capitalizeName(Issue issue, IssueResolutionAcceptor acceptor) {
    acceptor.accept(issue, 'Capitalize name', 'Capitalize the name.', 'upcase.png') [
        context |
        val xtextDocument = context.xtextDocument
        val firstLetter = xtextDocument.get(issue.offset, 1)
        xtextDocument.replace(issue.offset, 1, firstLetter.toUpperCase)
    ]
}
```

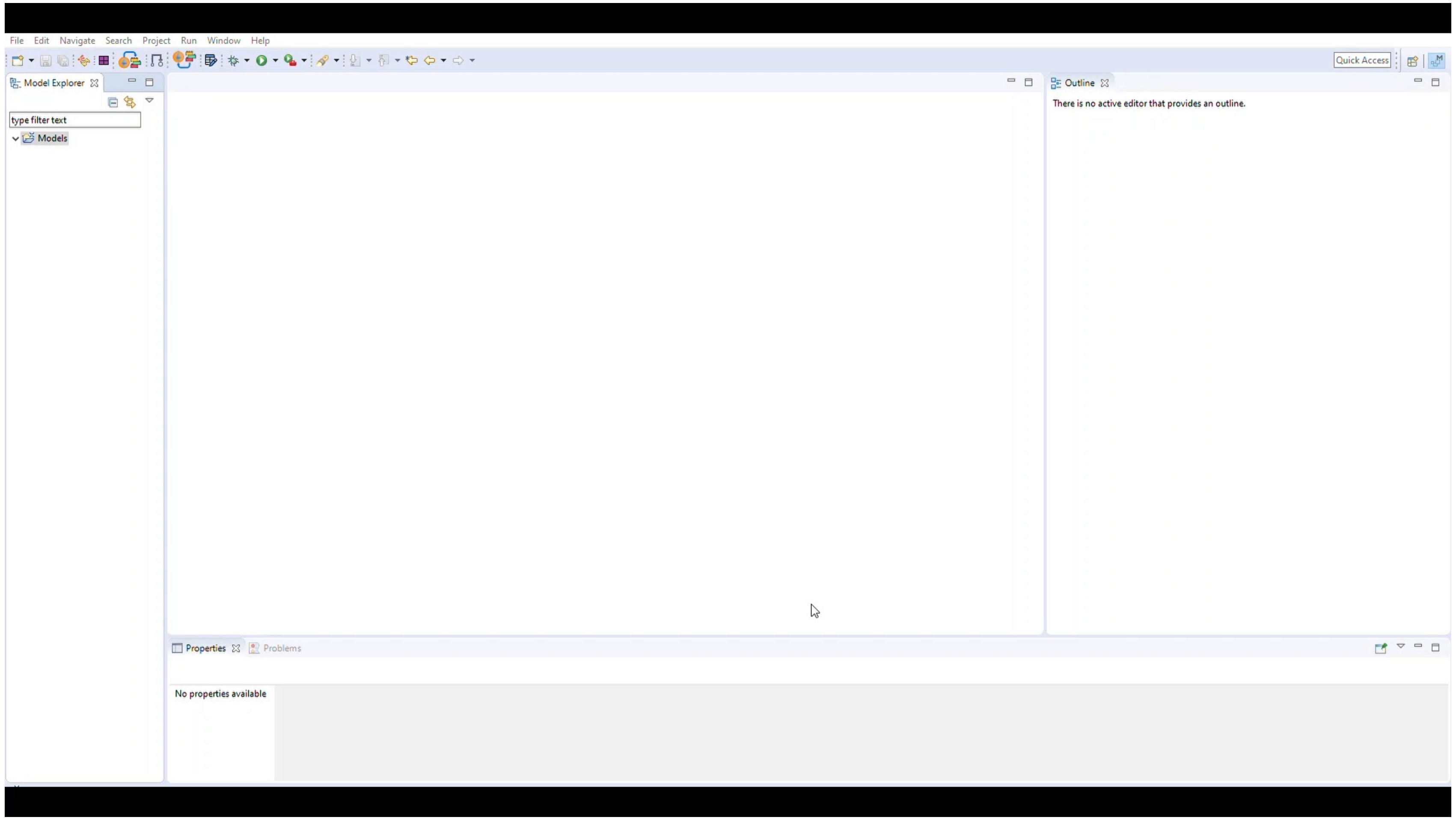
SmaC – a textual DSL for Smart Contract modeling

Features

- Coding Solidity Smart Contracts ➪ Smart Contract model
- Predefined data types: User & Company
- Facilities
  - Code completion
  - Syntax highlighting
  - Element tag description
  - Documentation
- Validation and quickfixes
- Structural pattern

SmaC – a textual DSL for Smart Contract modeling

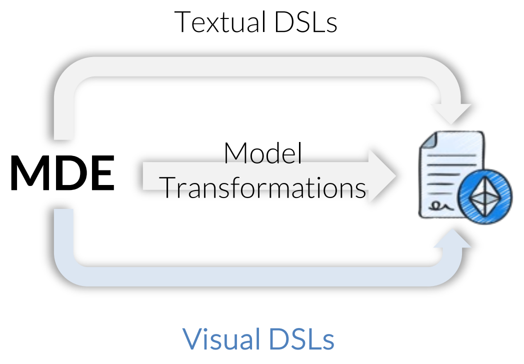
SmaC in action

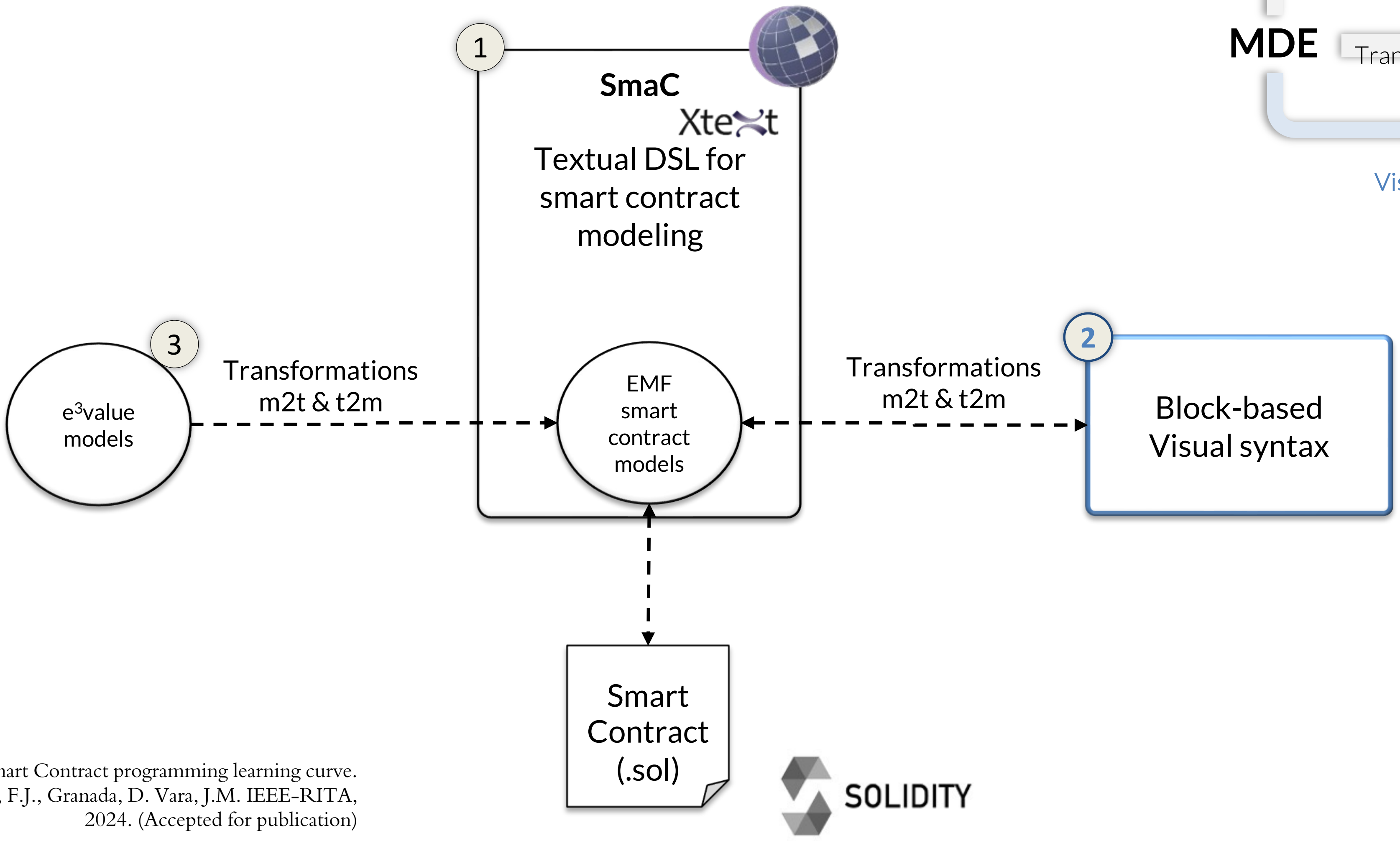


There is no active editor that provides an outline.

No properties available

Research Proposal (1.0) - SmaCly




1
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
3
e³value models
Transformations m2t & t2m
Transformations m2t & t2m
2
Block-based Visual syntax
Smart Contract (.sol)
SOLIDITY

A block-based web IDE to ease the Smart Contract programming learning curve. Gómez-Macías, C., Pérez-Blanco, F.J., Granada, D. Vara, J.M. IEEE-RITA, 2024. (Accepted for publication)

SmaCly – a block-based VDSL for Smart Contracts

SCRATCH

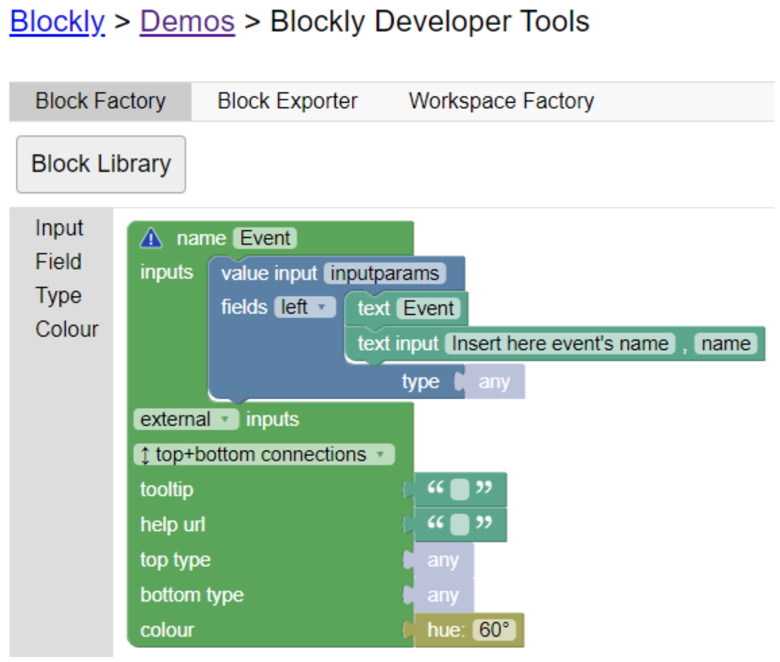
MIT APP INVENTOR

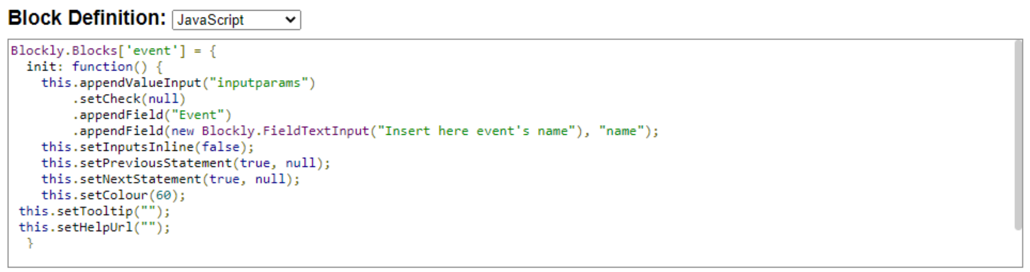
Greenfoot

Blockly

SmaCly – a block-based VDSL for Smart Contracts

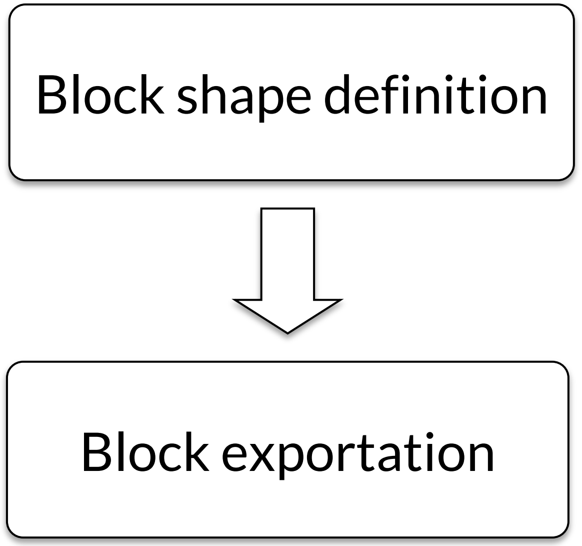
Block shape definition

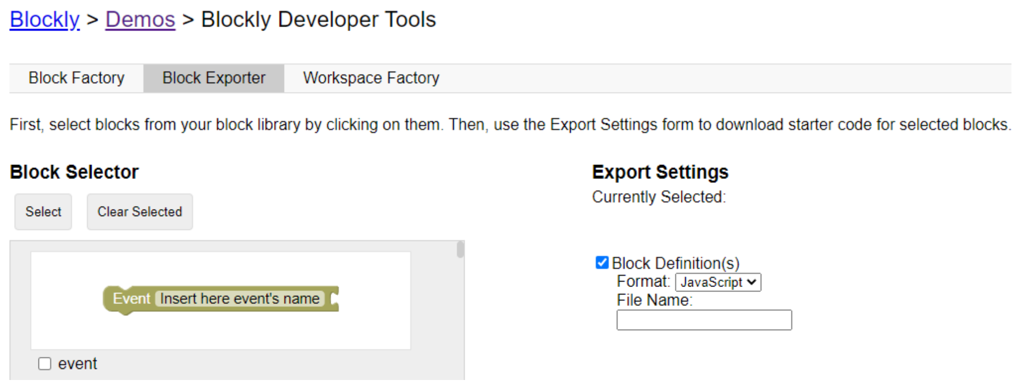



Block Definition: JavaScript

```
Blockly.Blocks['event'] = {
  init: function() {
    this.appendValueInput("inputparams")
        .setCheck(null)
        .appendField("Event")
        .appendField(new Blockly.FieldTextInput("Insert here event's name"), "name");
    this.setInputsInline(false);
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setColour(60);
 this.setTooltip("");
 this.setHelpUrl("");
  }
```

SmaCly – a block-based VDSL for Smart Contracts




Blockly > Demos > Blockly Developer Tools

First, select blocks from your block library by clicking on them. Then, use the Export Settings form to download starter code for selected blocks.

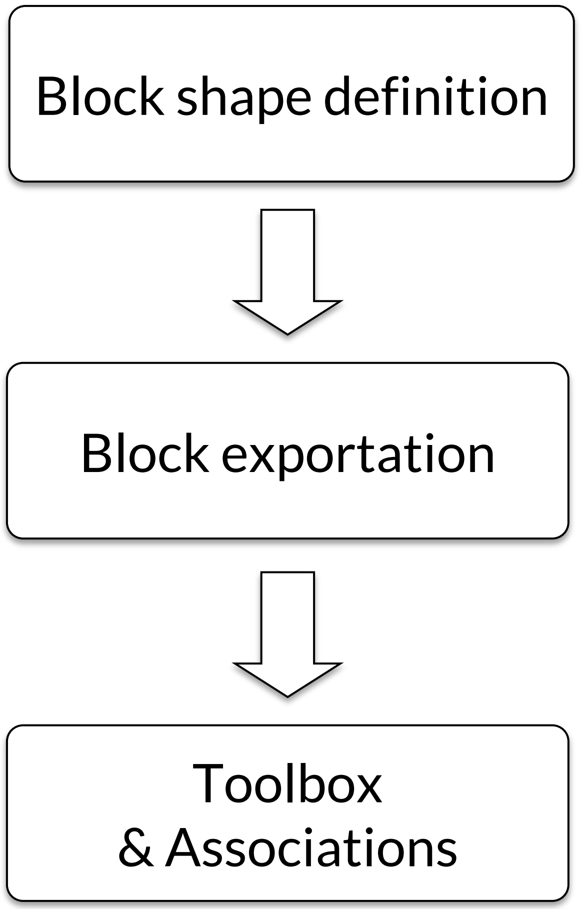
Block Selector

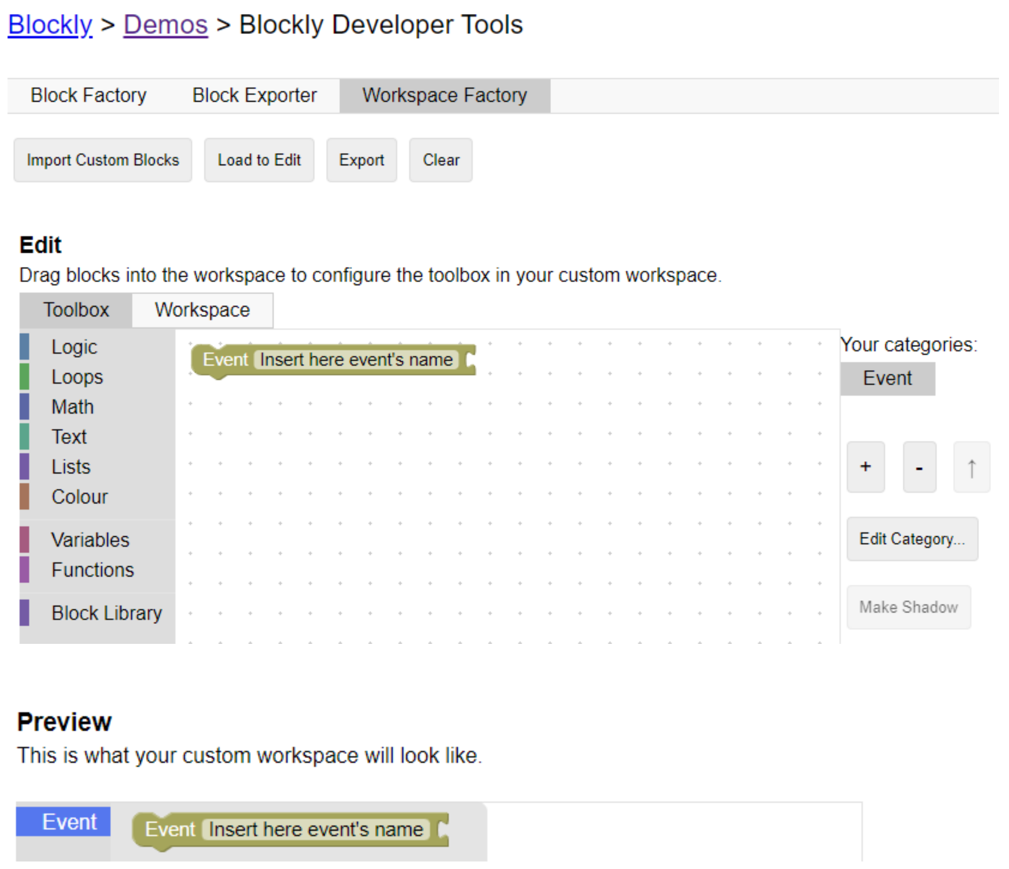
Export Settings

Currently Selected:

☑ Block Definition(s)
Format: JavaScript
File Name:

SmaCly – a block-based VDSL for Smart Contracts




Blockly > Demos > Blockly Developer Tools

Edit

Drag blocks into the workspace to configure the toolbox in your custom workspace.

Preview

This is what your custom workspace will look like.

SmaCly – a block-based VDSL for Smart Contracts

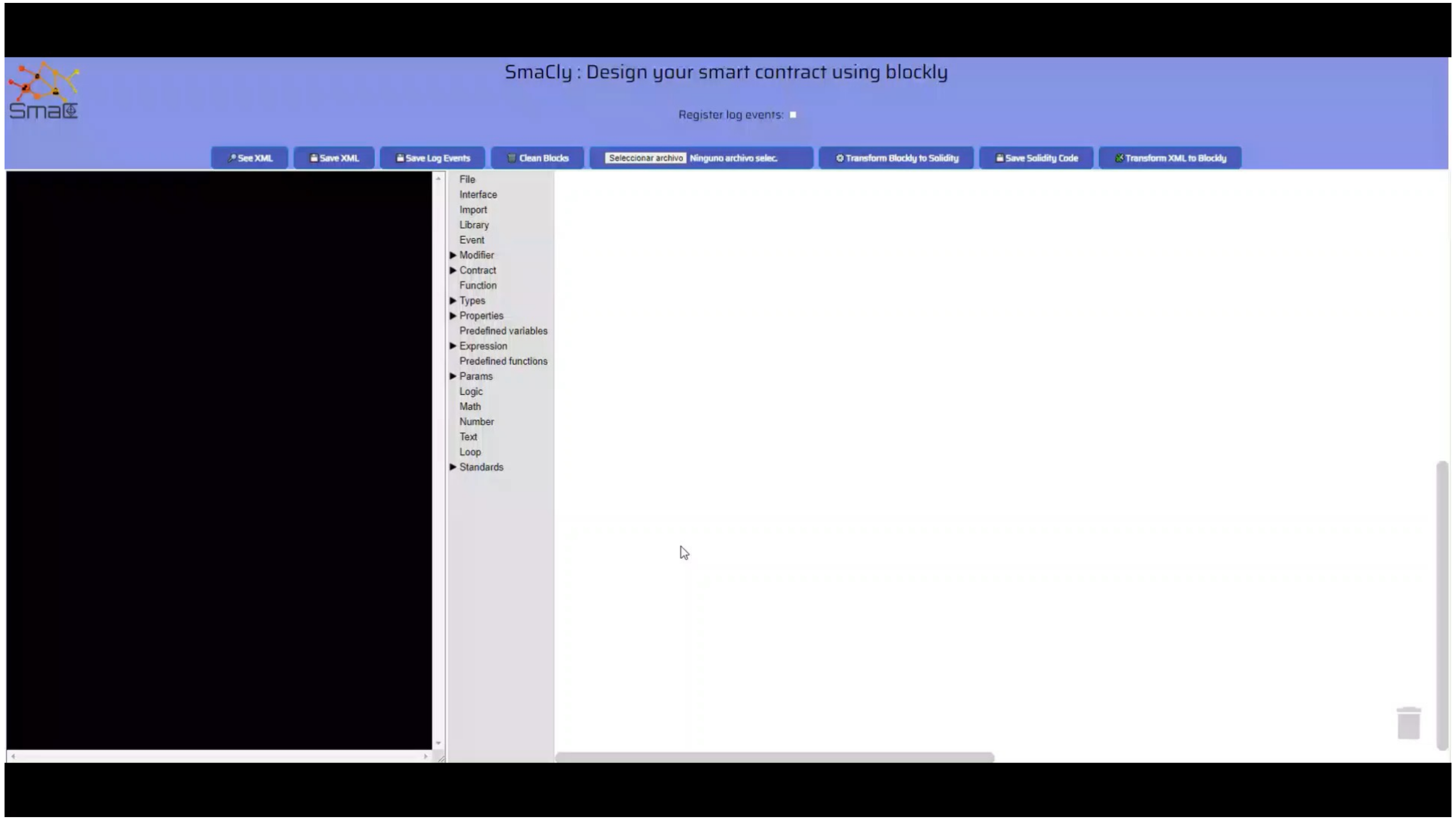
Features

- Visual programming of Solidity Smart contracts
- Predefined data types: User & Company
- Facilities: model element description and documentation
- Predefined templates (Fungible & Non fungible tokens)
- Inherits SmaC's structural pattern
- Import and export mechanisms (models, running-code, XML formats)

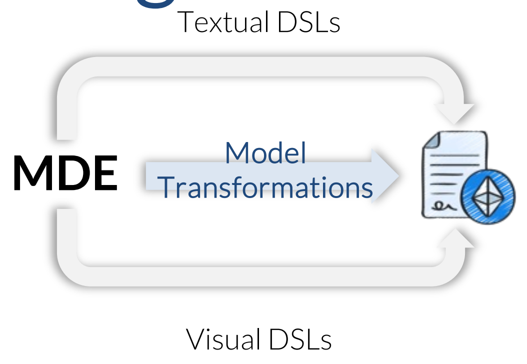
SmaCly supports the creation of models based on the abstract syntax defined for SmaC

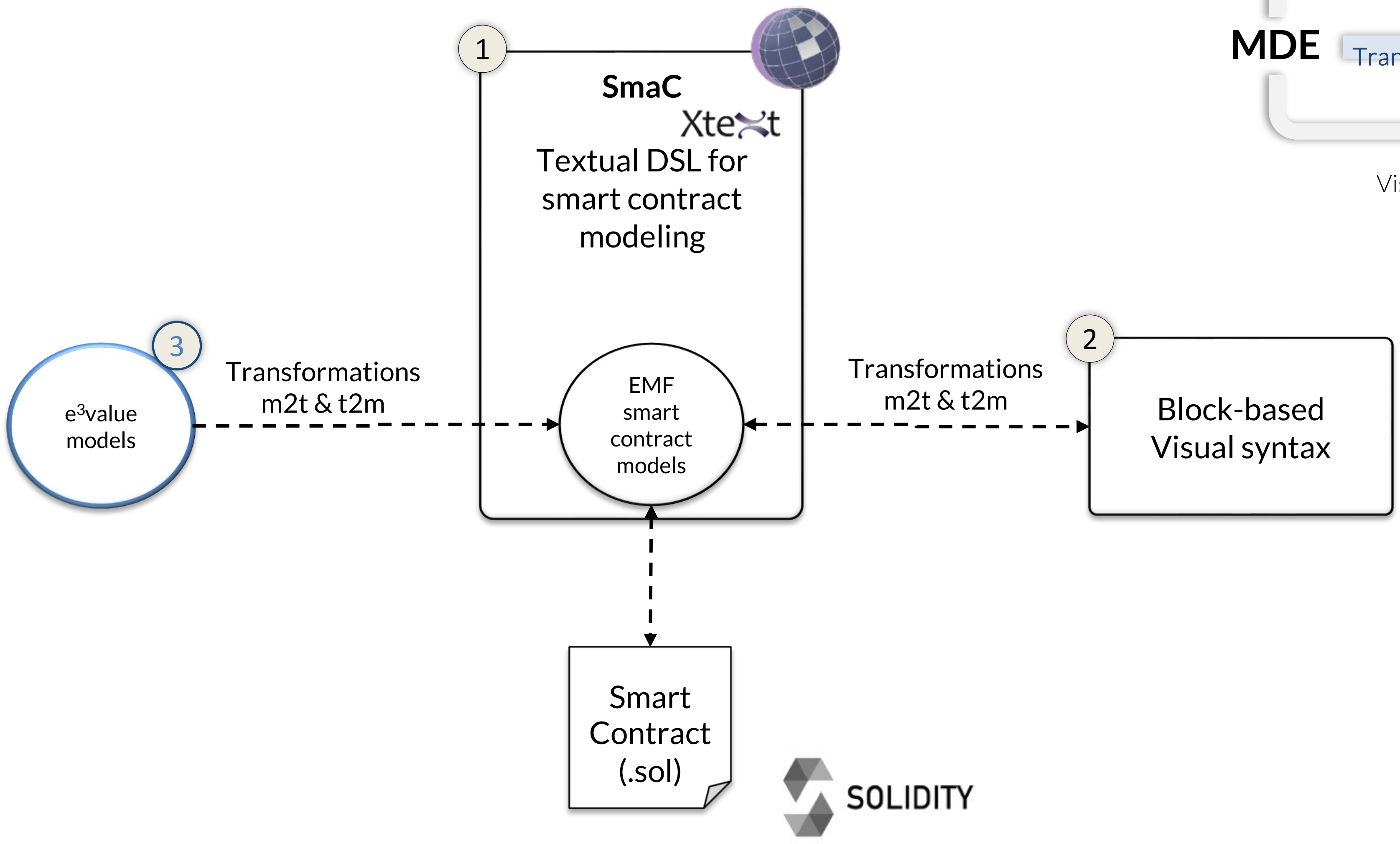
SmaCly – a block-based VDSL for Smart Contracts

SmaCly in action



Research Proposal (1.0) – Technological Bridges




1
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
3
e³value models
Transformations m2t & t2m
2
Transformations m2t & t2m
Block-based Visual syntax
Smart Contract (.sol)
SOLIDITY

Business (Process) Modeling

Canvas

e³value

Service Blueprint

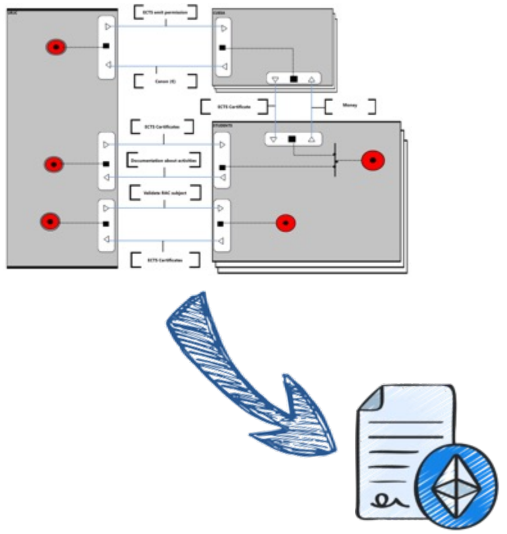
PCN

BPMN

Business (Process) Modeling

Canvas

e³value



Service Blueprint

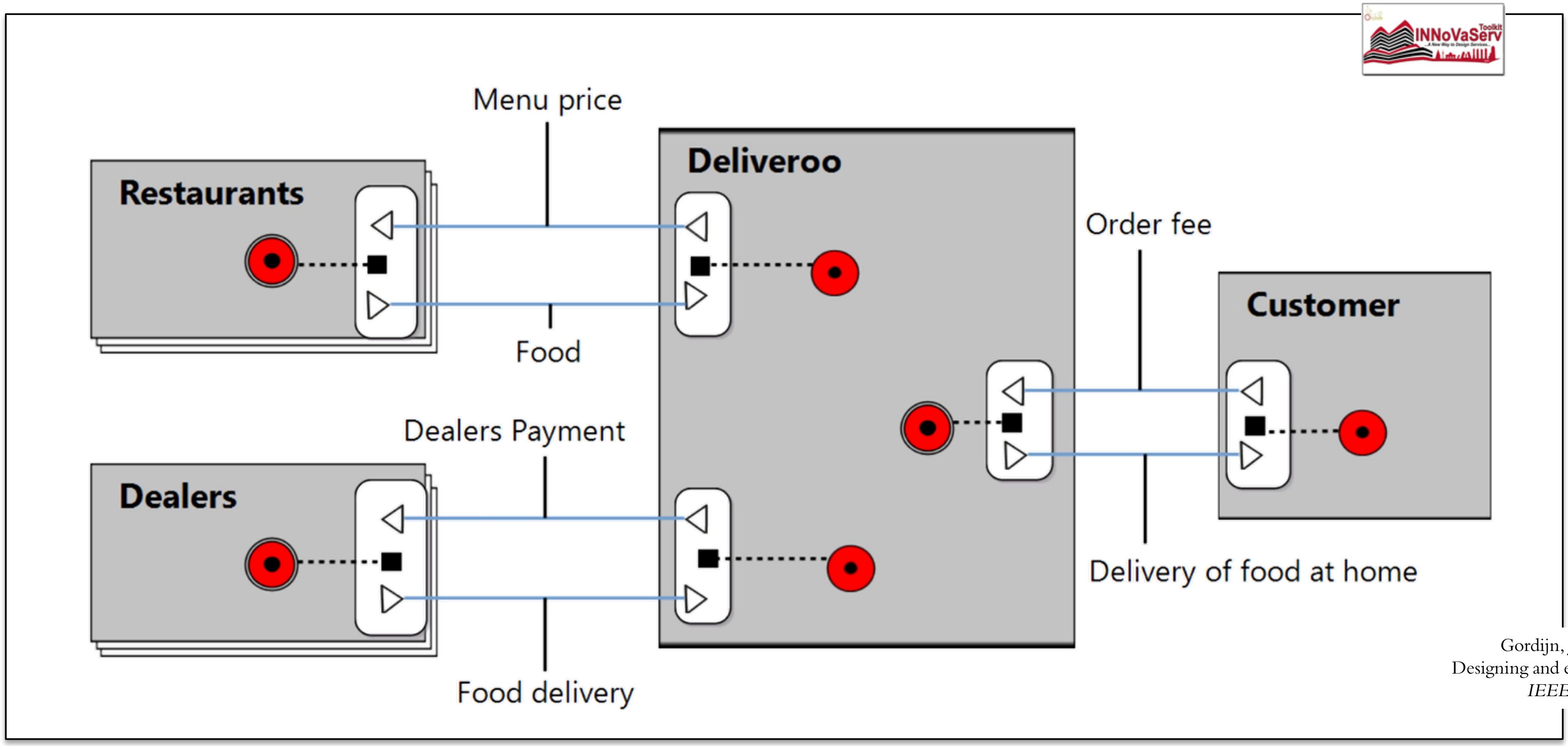
PCN

BPMN

e^{3}value in a nutshell

Business modeling notation

- Focused on representing the value interchanges between the different actors involved in the provision of a service.

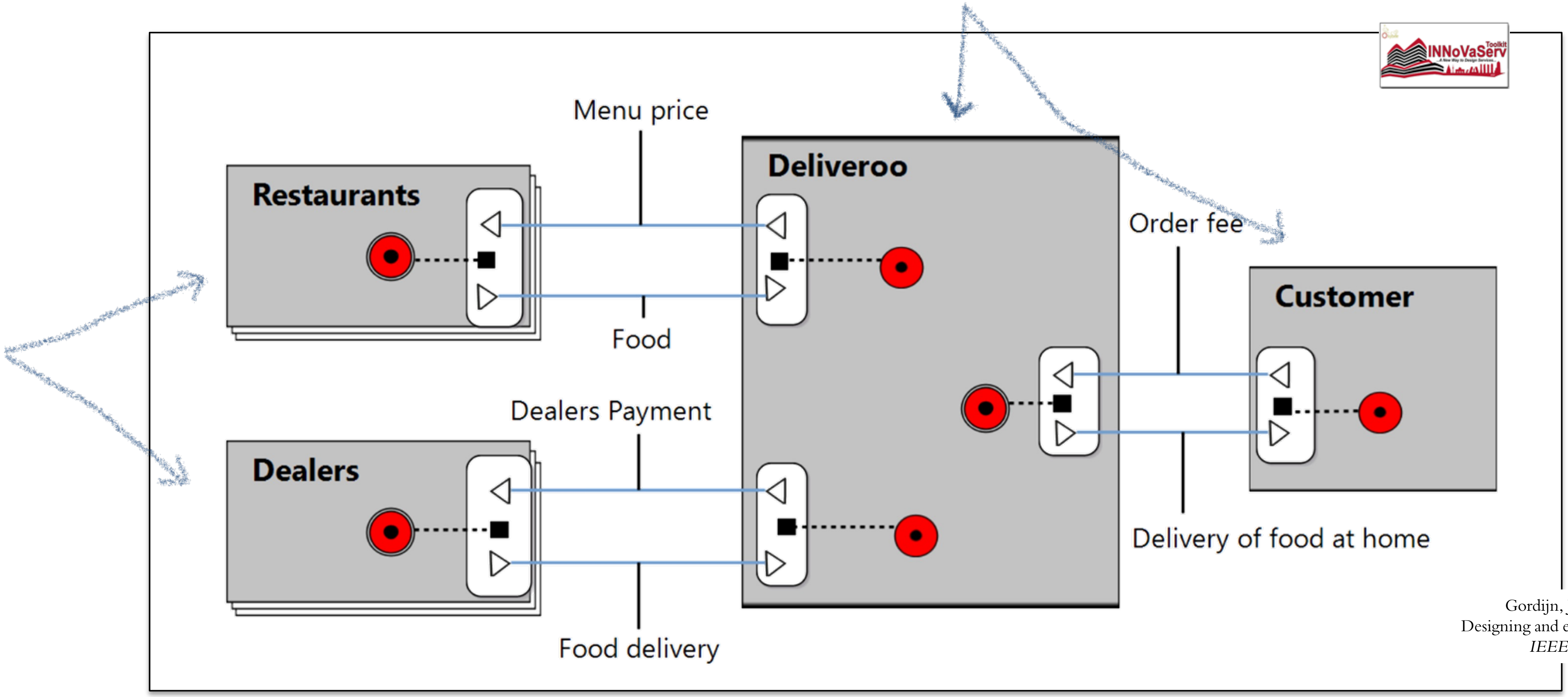


Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

e^{3}value in a nutshell

Actors / Market Segments

- An entity that carries out value activities that allow him/her/it to increase … profit or utility

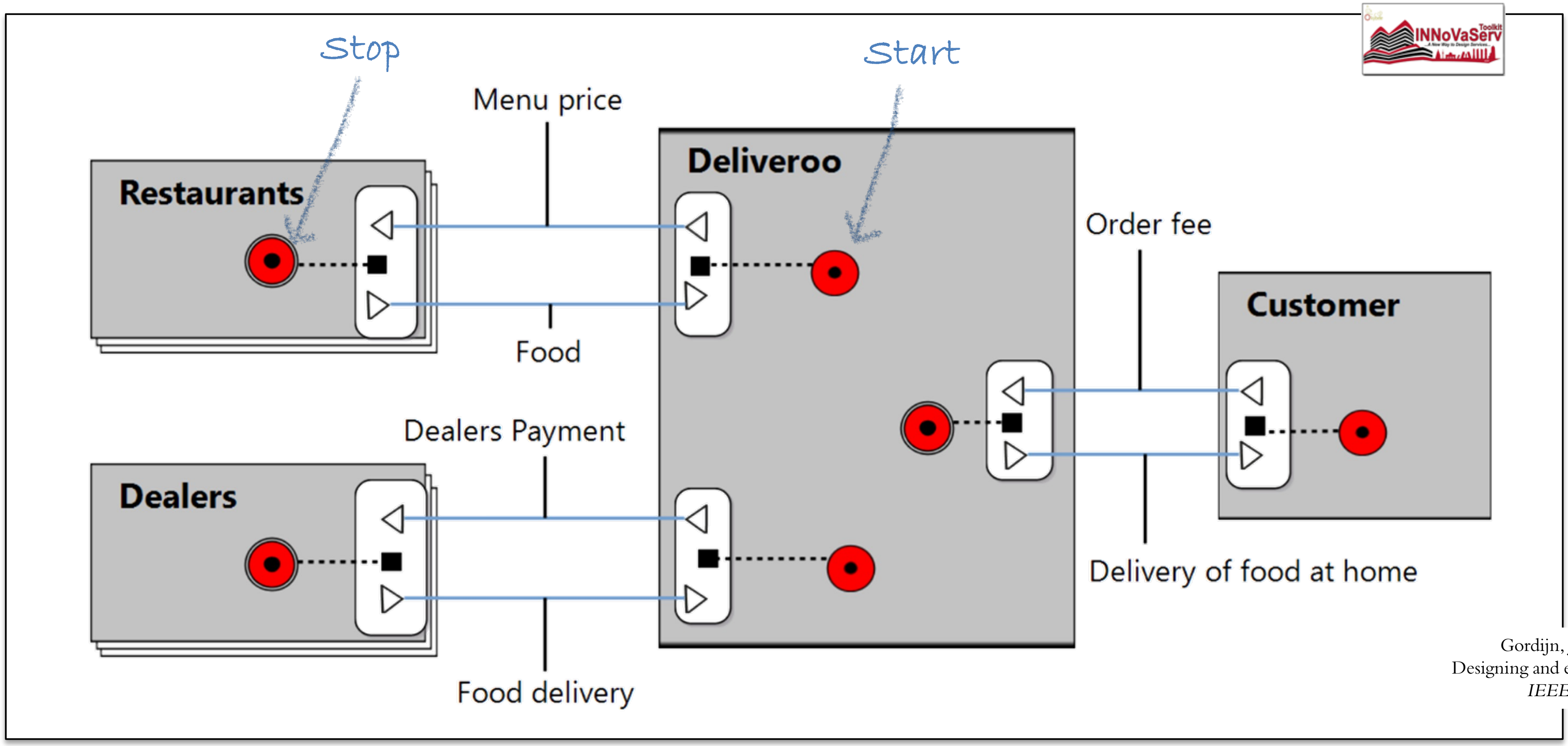


Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

e^{3}value in a nutshell

Stimulus

- Events caused by an actor, trigger come value exchange.
- Two types: Start stimulus (User Need) y Stop stimulus Border Item.

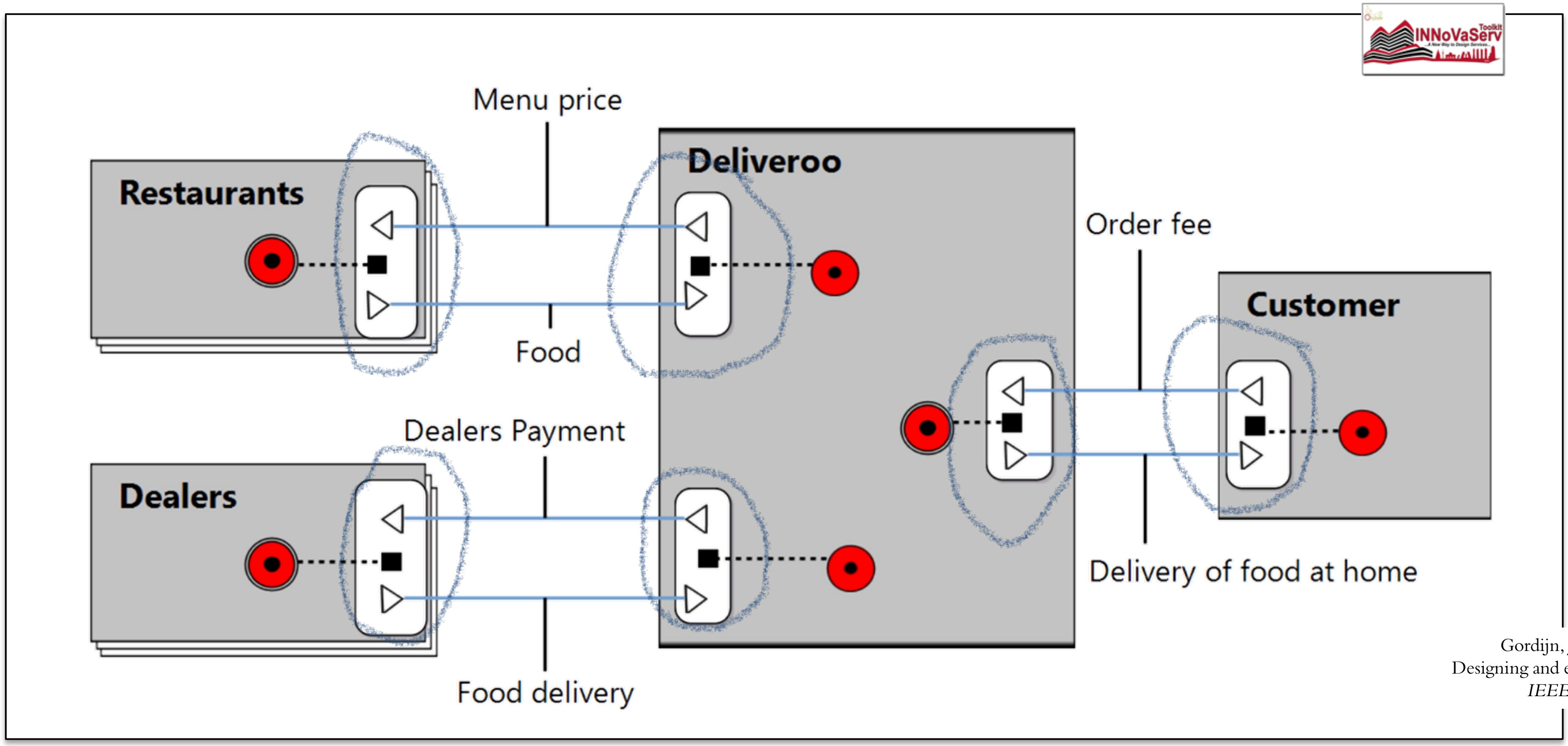


Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

e³value in a nutshell

Value Interface

- Group the ports through which the actor is willing to make value interchanges
- A form of representation of economic reciprocity of value between actors

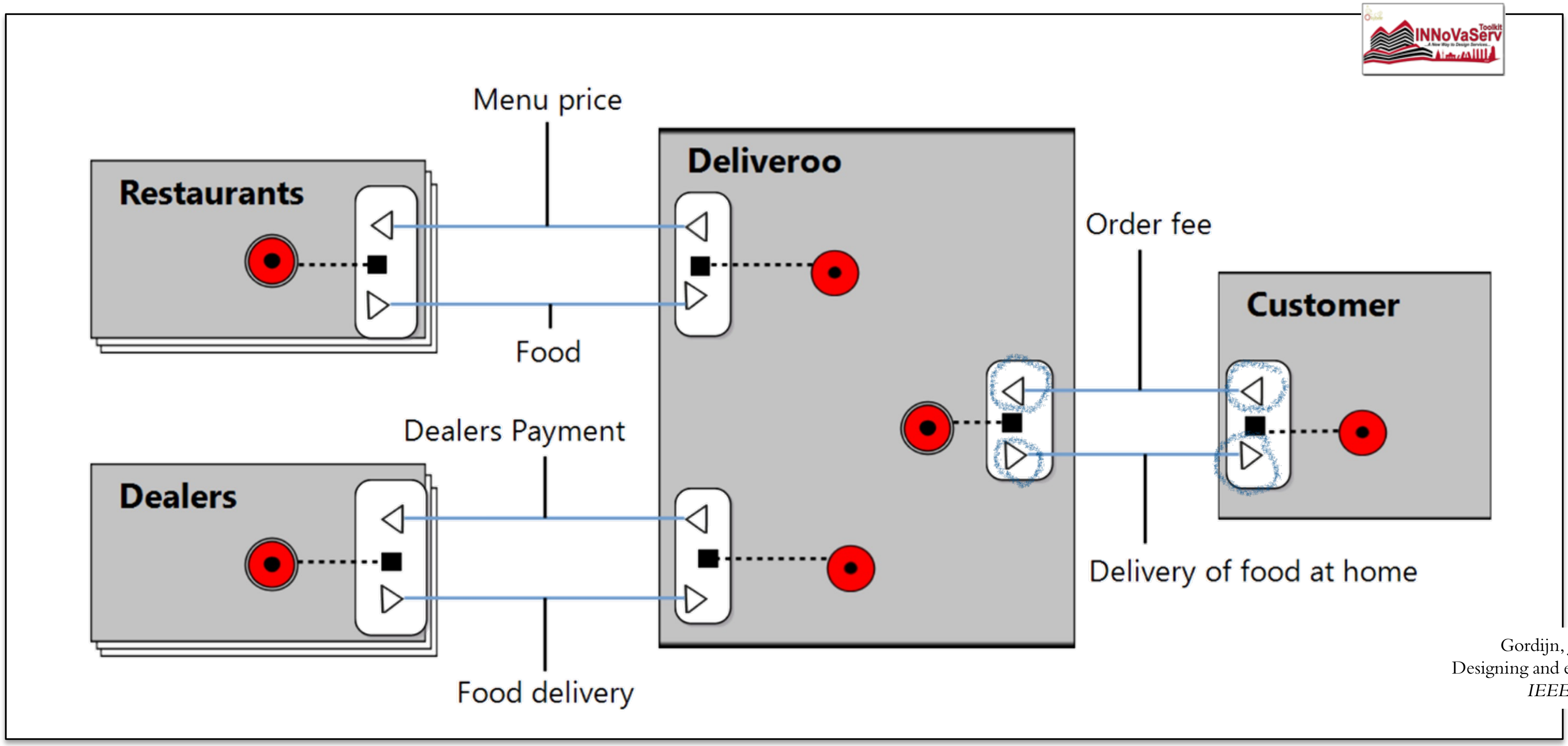


Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

e^{3}value in a nutshell

Value Ports

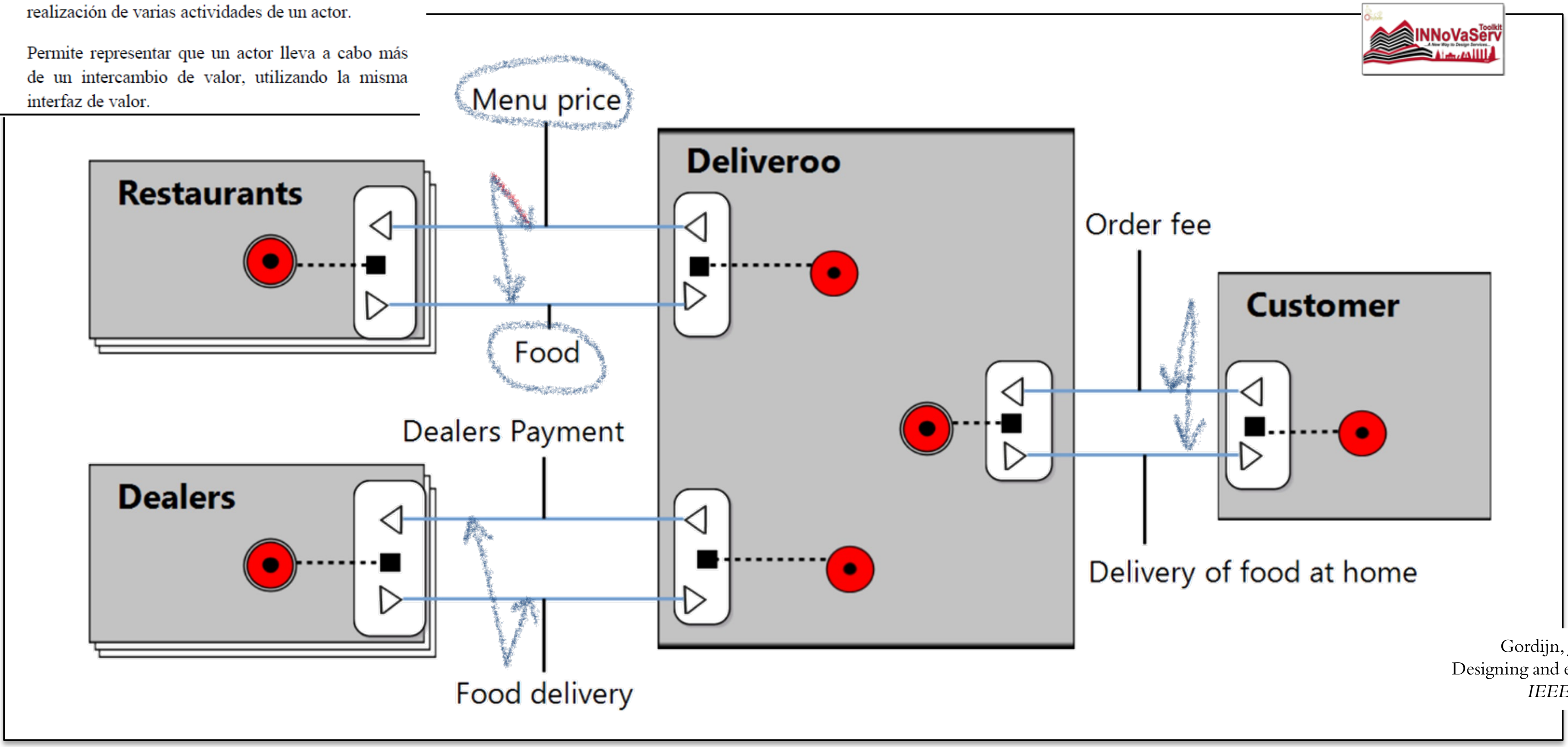
- Used by an actor to request **value objects** to or from its environment (directional)



Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

e^{3}value in a nutshell

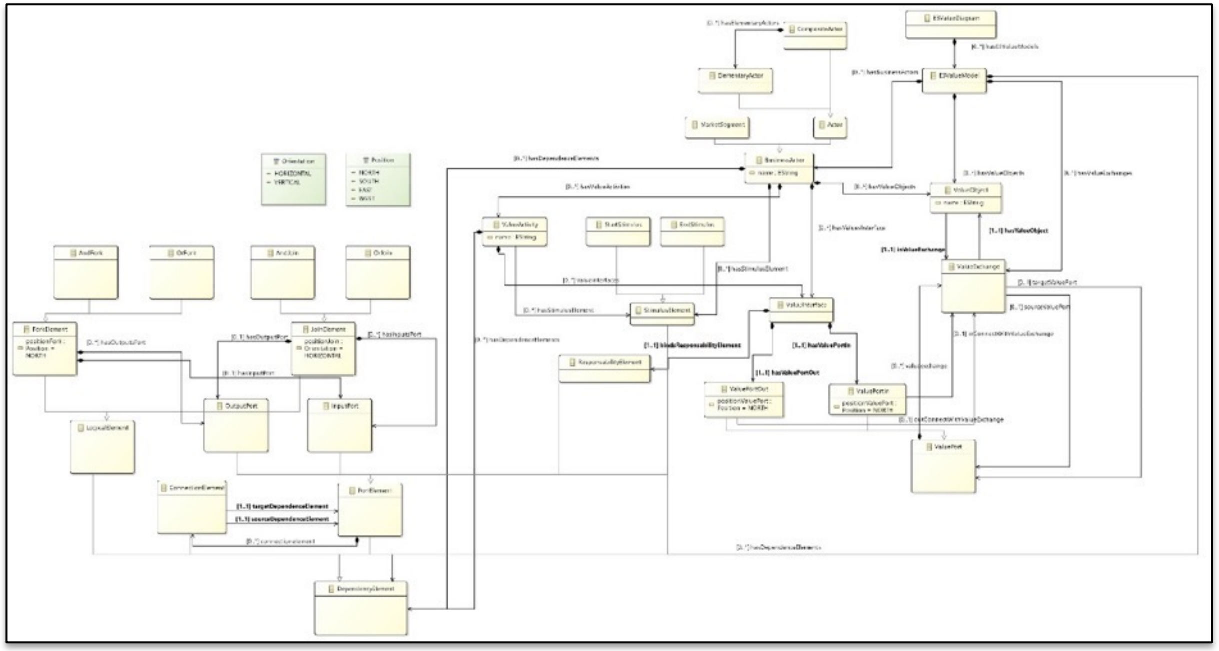
Elemento	*Representación*	*Descripción*
Intercambio de valor	▬	Representa los canales a través de los que se lleva a cabo un intercambio de objetos de valor.
Objeto de valor	[...]	Representa los bienes, productos o servicios intercambiados.
Conexión	----	Permiten enlazar los diferentes elementos contenidos en los actores.
Dependencia AND	⏀	Permite representar el operador booleano AND cuando se combinan varias actividades en un actor.
Dependencia OR	⏀	Permite representar el operador booleano OR en la realización de varias actividades de un actor.
Implosión / Explosión	1 ● ‖ ● 1	Permite representar que un actor lleva a cabo más de un intercambio de valor, utilizando la misma interfaz de valor.



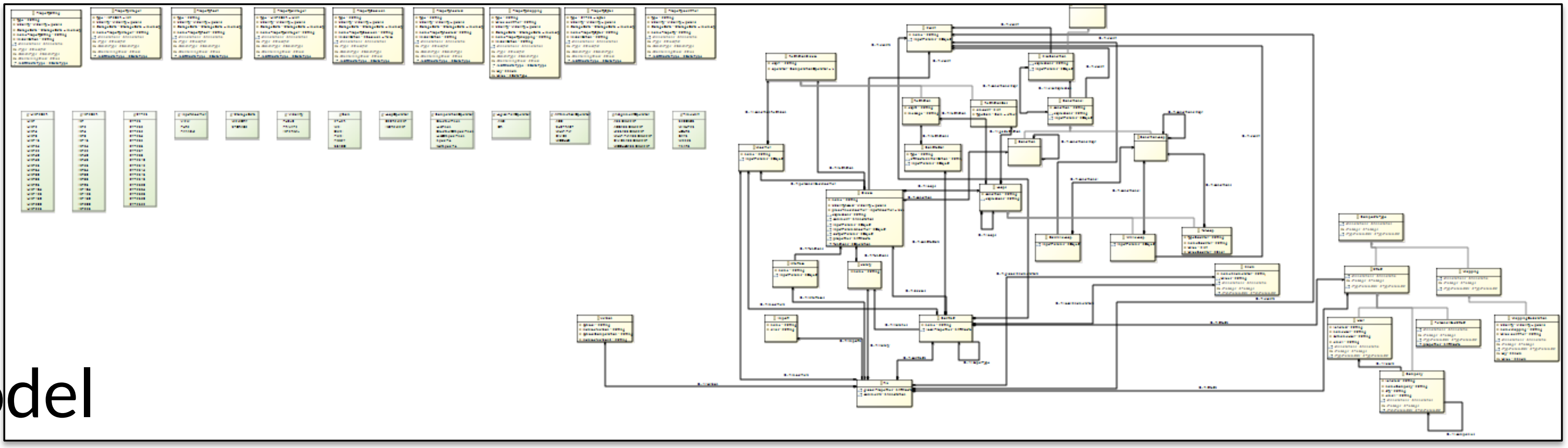
Gordijn, J., & Akkermans, H. (2001). Designing and evaluating e-business models. *IEEE intelligent Systems*, (4), 11-17.

Abstract Syntaxes – Concepts

e^{3}value Metamodel

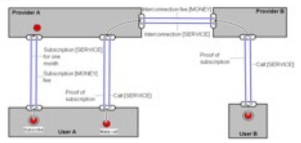


SmaC Metamodel



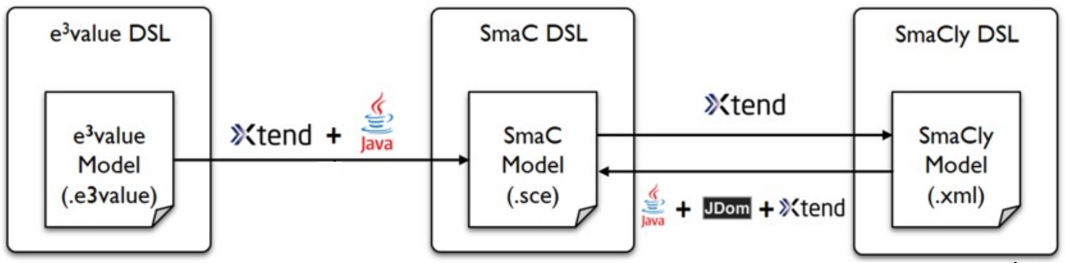
Correspondences Analysis

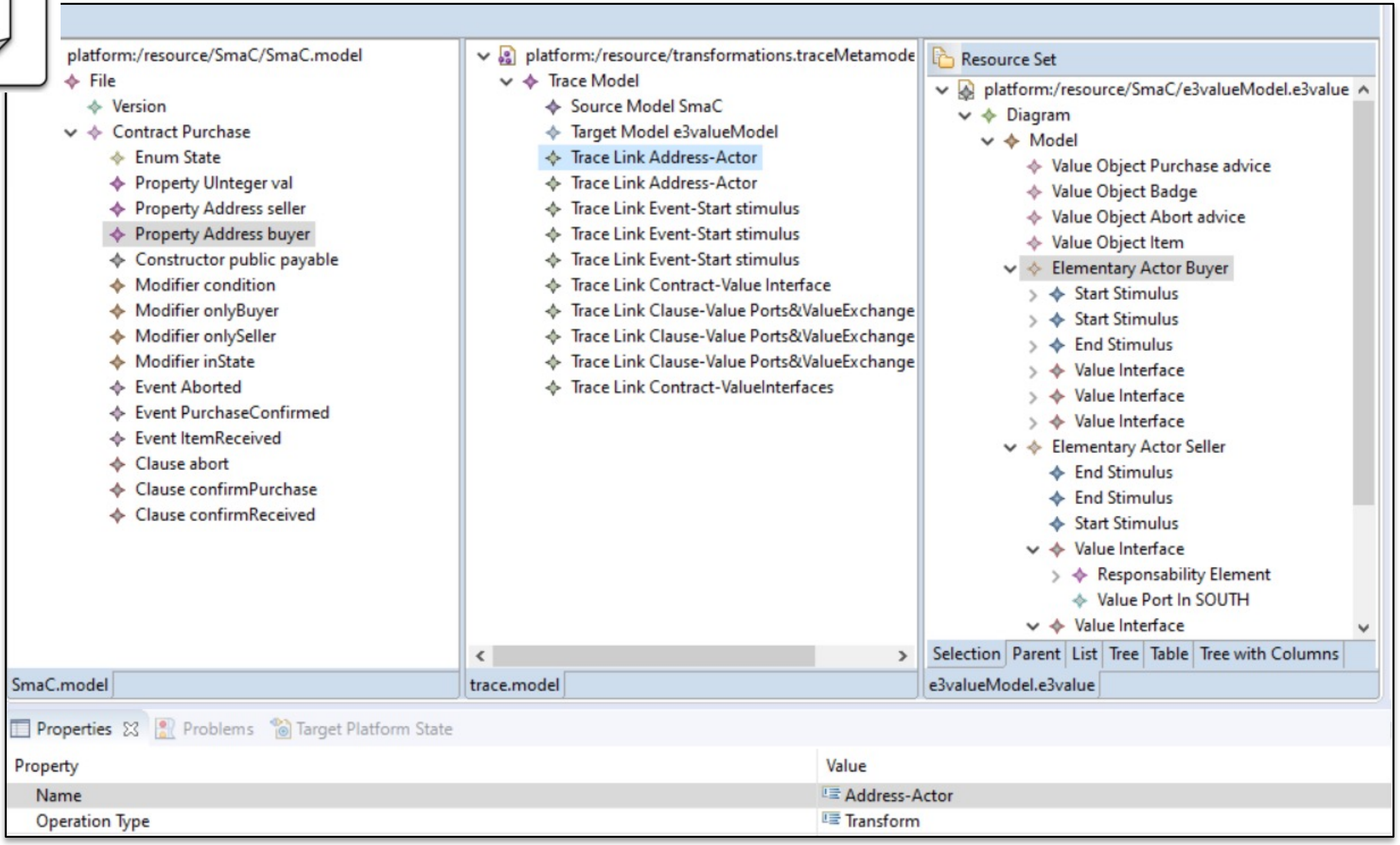
Address / User (Personalized Struct)	⟷	Actors
Address Array / Company (Personalized Struct)	⟷	Market segment
Cryptocoins, Services, Assets, Advices	⟷	Value Object
Events	⟷	Stimulus
Smart contract	<----->	Value Interface
Functions	⟷	Value Port, Value Transfer



⟷ Match
<----> Partial Match

Model Weaving for correspondences visualization





platform:/resource/SmaC/SmaC.model
- File
  - Version
  - Contract Purchase
    - Enum State
    - Property UInteger val
    - Property Address seller
    - Property Address buyer
    - Constructor public payable
    - Modifier condition
    - Modifier onlyBuyer
    - Modifier onlySeller
    - Modifier inState
    - Event Aborted
    - Event PurchaseConfirmed
    - Event ItemReceived
    - Clause abort
    - Clause confirmPurchase
    - Clause confirmReceived

SmaC.model

platform:/resource/transformations.traceMetamode
- Trace Model
  - Source Model SmaC
  - Target Model e3valueModel
  - Trace Link Address-Actor
  - Trace Link Address-Actor
  - Trace Link Event-Start stimulus
  - Trace Link Event-Start stimulus
  - Trace Link Event-Start stimulus
  - Trace Link Contract-Value Interface
  - Trace Link Clause-Value Ports&ValueExchange
  - Trace Link Clause-Value Ports&ValueExchange
  - Trace Link Clause-Value Ports&ValueExchange
  - Trace Link Contract-ValueInterfaces

trace.model

Resource Set
- platform:/resource/SmaC/e3valueModel.e3value
  - Diagram
    - Model
      - Value Object Purchase advice
      - Value Object Badge
      - Value Object Abort advice
      - Value Object Item
      - Elementary Actor Buyer
        - Start Stimulus
        - Start Stimulus
        - End Stimulus
        - Value Interface
        - Value Interface
        - Value Interface
      - Elementary Actor Seller
        - End Stimulus
        - End Stimulus
        - Start Stimulus
        - Value Interface
          - Responsability Element
          - Value Port In SOUTH
        - Value Interface

Selection | Parent | List | Tree | Table | Tree with Columns

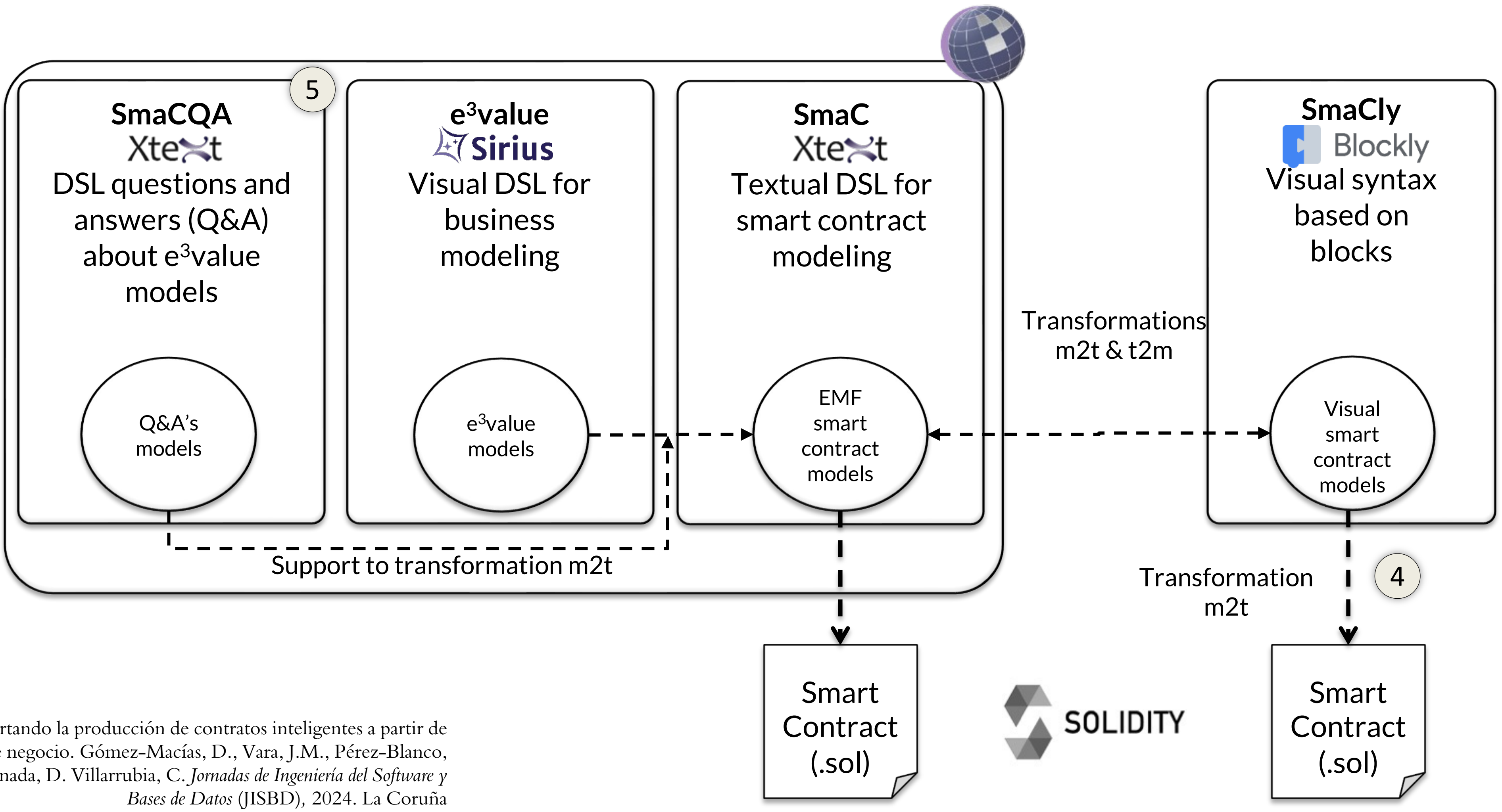
e3valueModel.e3value

Properties | Problems | Target Platform State

Property	Value
Name	Address-Actor
Operation Type	Transform

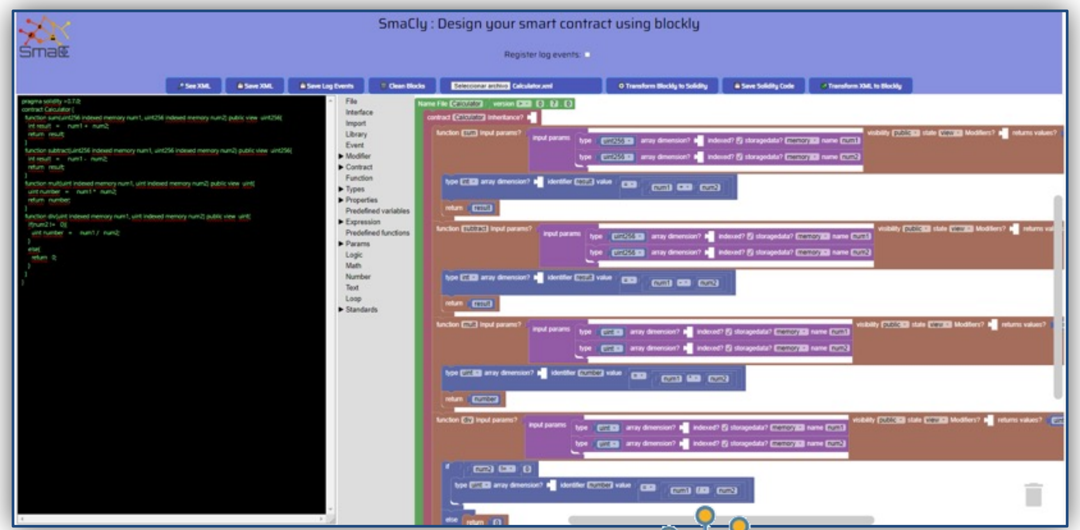
Agenda

Research Proposal (2.0)

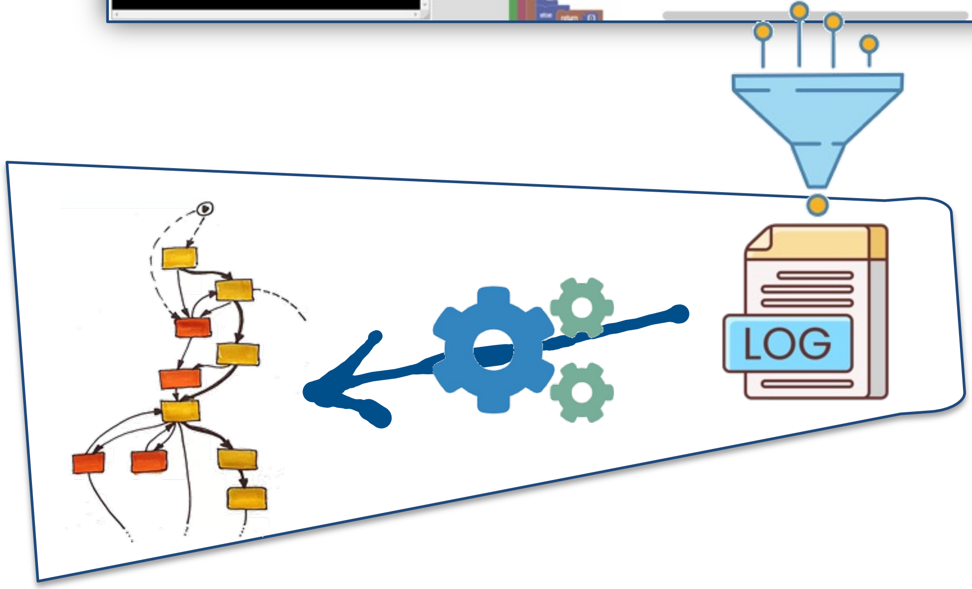

5
SmaCQA
Xtext
DSL questions and answers (Q&A) about e3value models
Q&A's models
e3value
Sirius
Visual DSL for business modeling
e3value models
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
Support to transformation m2t
Transformations m2t & t2m
SmaCly
Blockly
Visual syntax based on blocks
Visual smart contract models
Transformation m2t
4
Smart Contract (.sol)
SOLIDITY
Smart Contract (.sol)

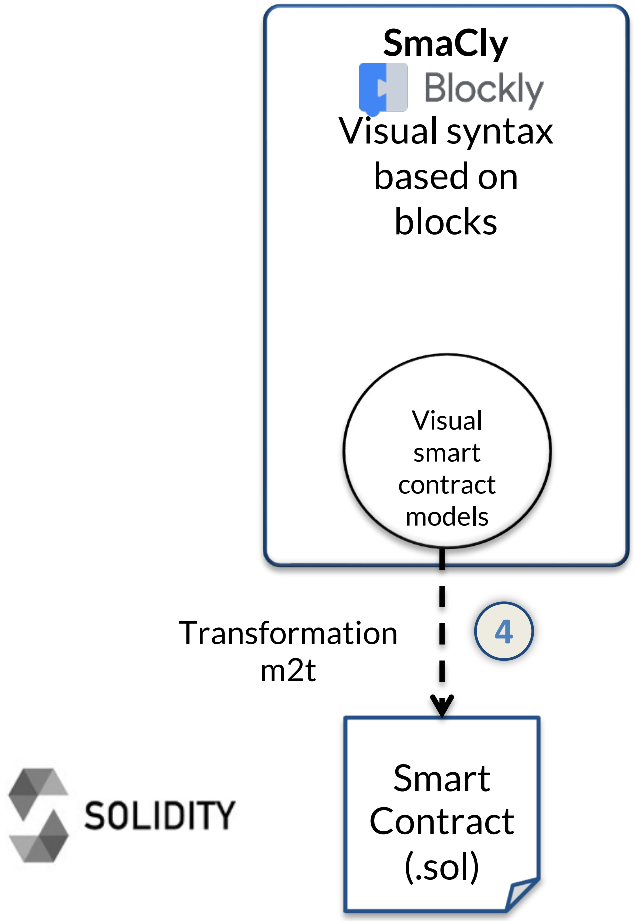
Soportando la producción de contratos inteligentes a partir de modelos de negocio. Gómez-Macías, D., Vara, J.M., Pérez-Blanco, F.J., Granada, D. Villarrubia, C. *Jornadas de Ingeniería del Software y Bases de Datos* (JISBD), 2024. La Coruña

Research Proposal (2.0) – SmaCly Addons

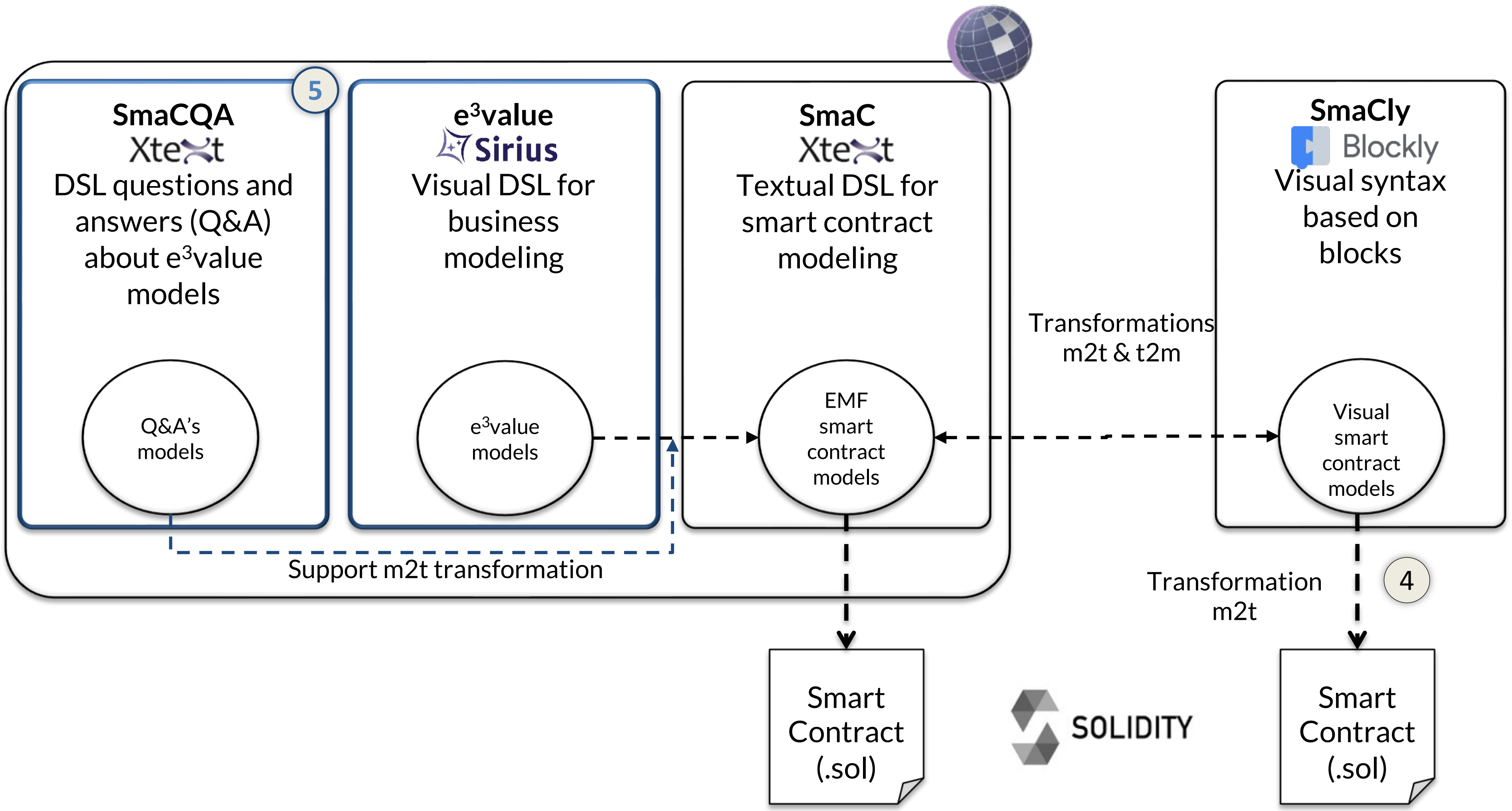


```
pragma solidity ^0.7.0;
contract Calculator {
  function sum(uint256 indexed memory num1, uint256 indexed memory num2) public view  uint256{
    int result  =    num1 +  num2;
    return  result;
  }
  function subtract(uint256 indexed memory num1, uint256 indexed memory num2) public view  uint256{
    int result  =    num1 -  num2;
    return  result;
  }
  function mult(uint indexed memory num1, uint indexed memory num2) public view  uint{
    uint number  =    num1 *  num2;
    return  number;
  }
  function div(uint indexed memory num1, uint indexed memory num2) public view  uint{
    if(num2 !=  0){
      uint number  =    num1 /  num2;
    }
    else{
      return  0;
    }
  }
}
```





Research Proposal (2.0) – SmaCQA



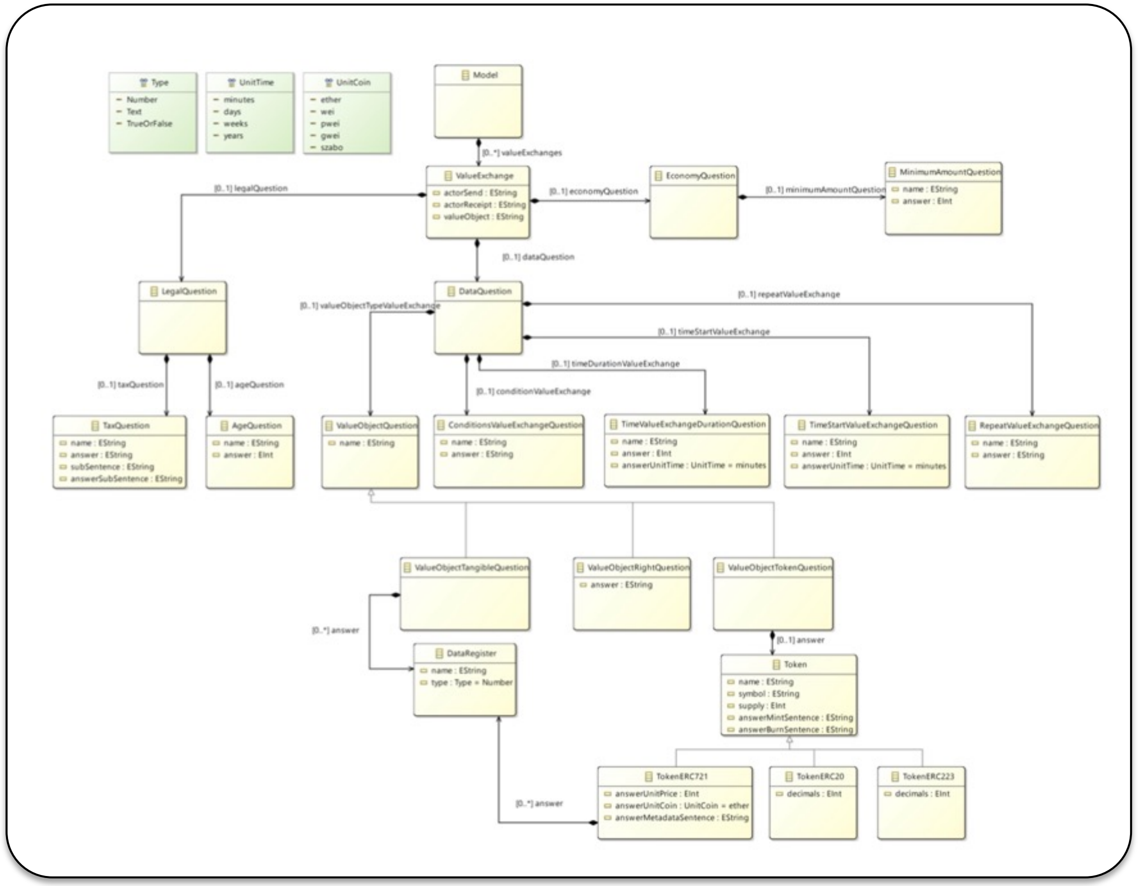
5
SmaCQA
Xtext
DSL questions and answers (Q&A) about e3value models
Q&A's models
e3value
Sirius
Visual DSL for business modeling
e3value models
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
Support m2t transformation
Transformations m2t & t2m
SmaCly
Blockly
Visual syntax based on blocks
Visual smart contract models
Transformation m2t
4
Smart Contract (.sol)
SOLIDITY
Smart Contract (.sol)

SmaCQA – a textual DSL for data gathering

Features

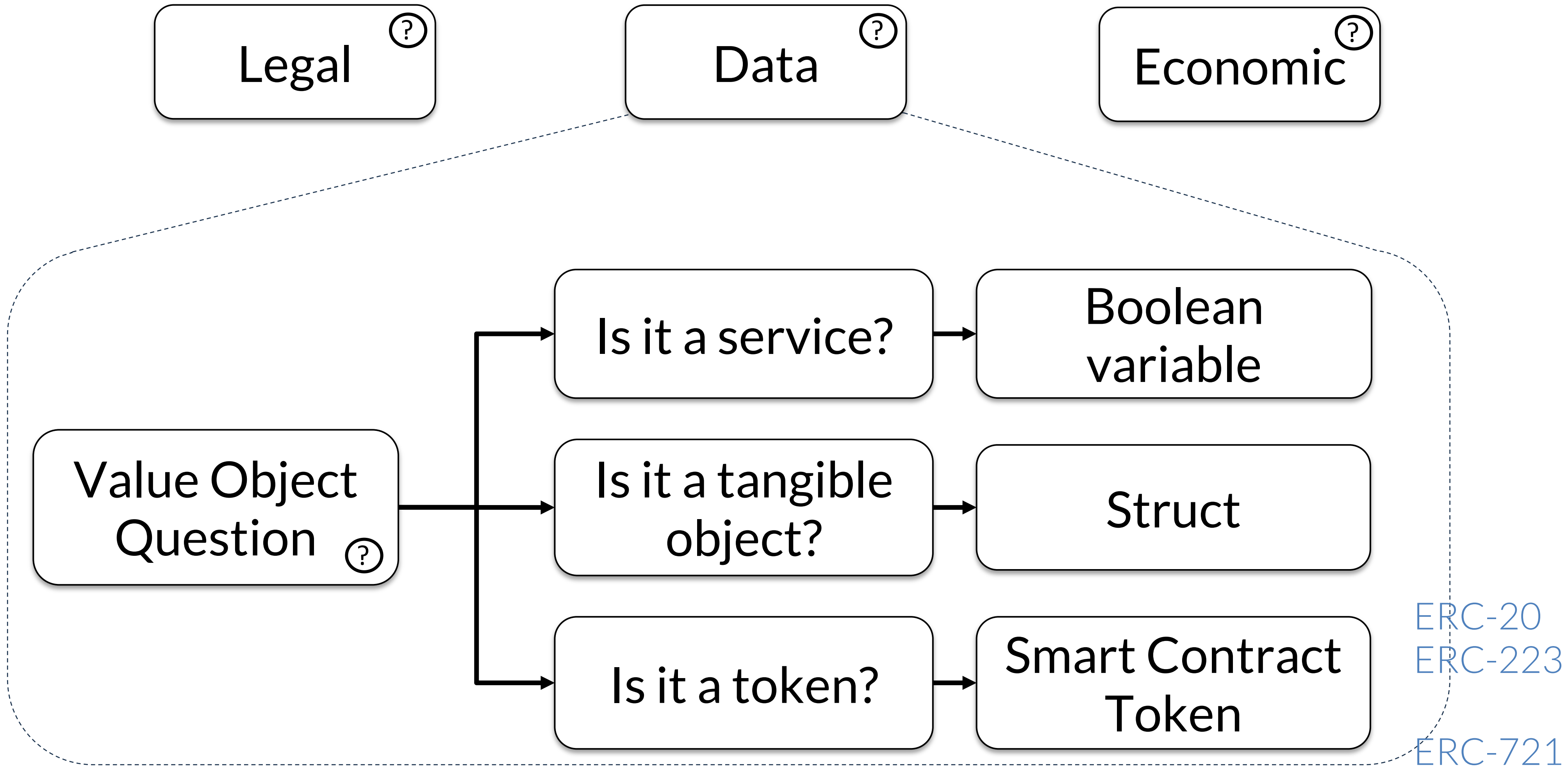
Xtext

- Additional information for e3value2SmaC mapping
- Facilities:
  - Syntax highlighting
  - Element tag description
  - Documentation
- High level design ~ Comprehesion
- Validation and quickfix
- HTML Export



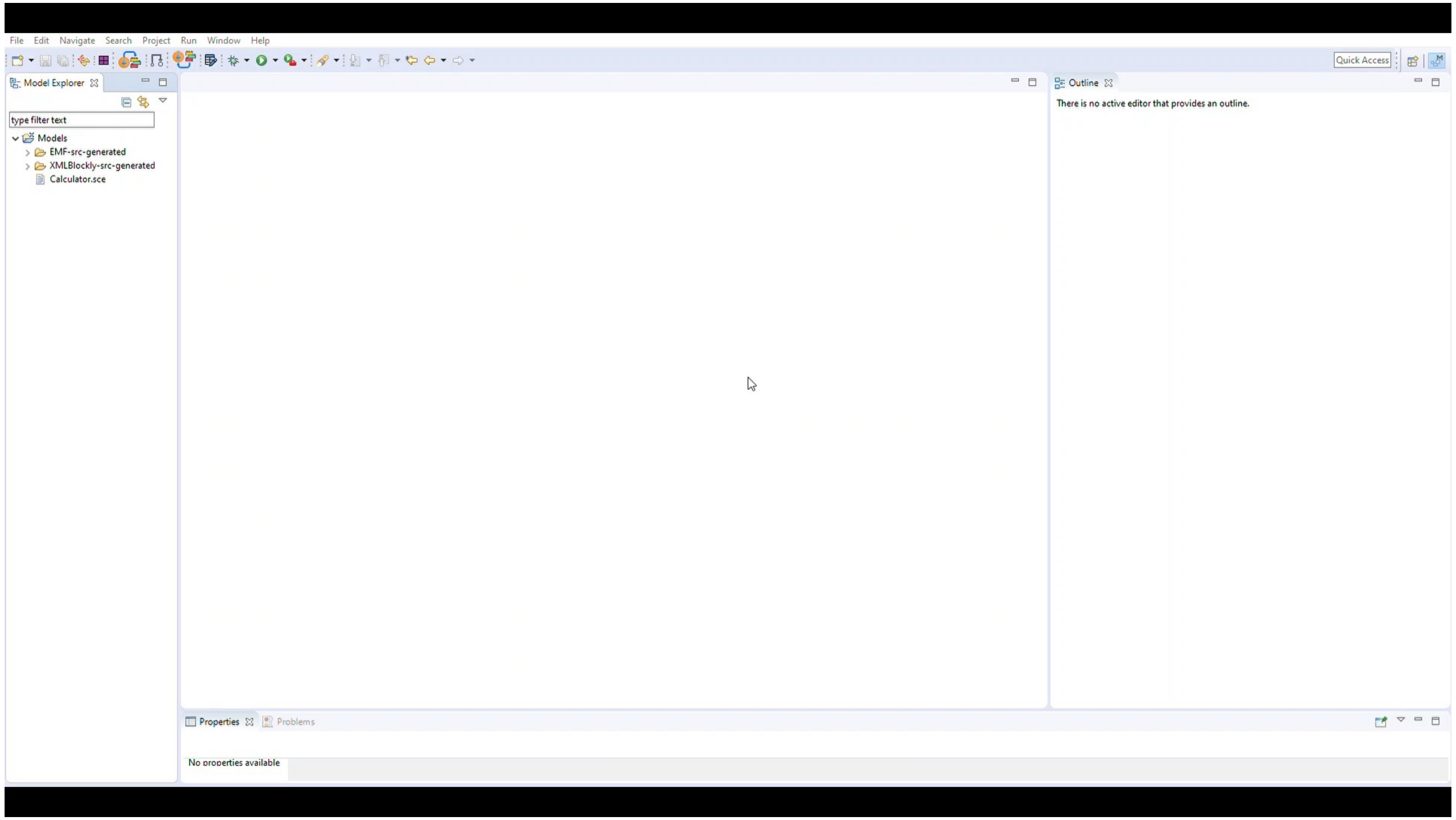
SmaCQA – questions supported

Supported questions



Running SmaCQA

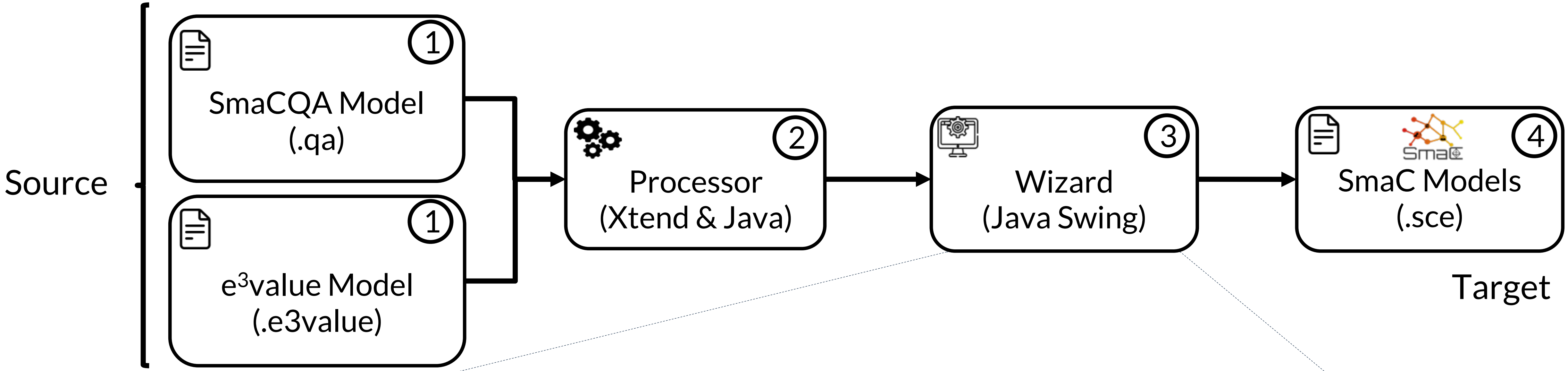
[SmaCQA in action]



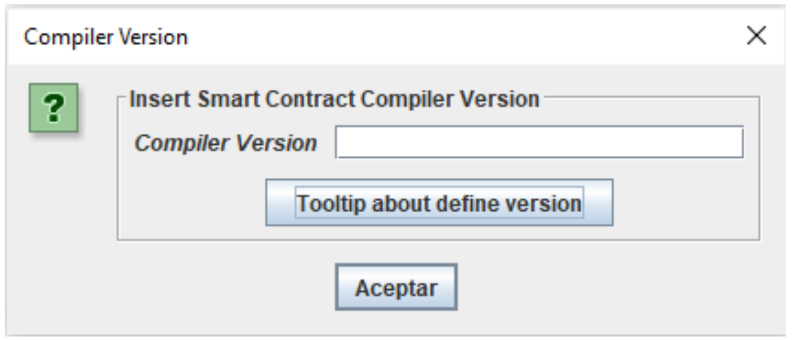
There is no active editor that provides an outline.

No properties available

From e^{3}value to SmaC models with SmaCQA

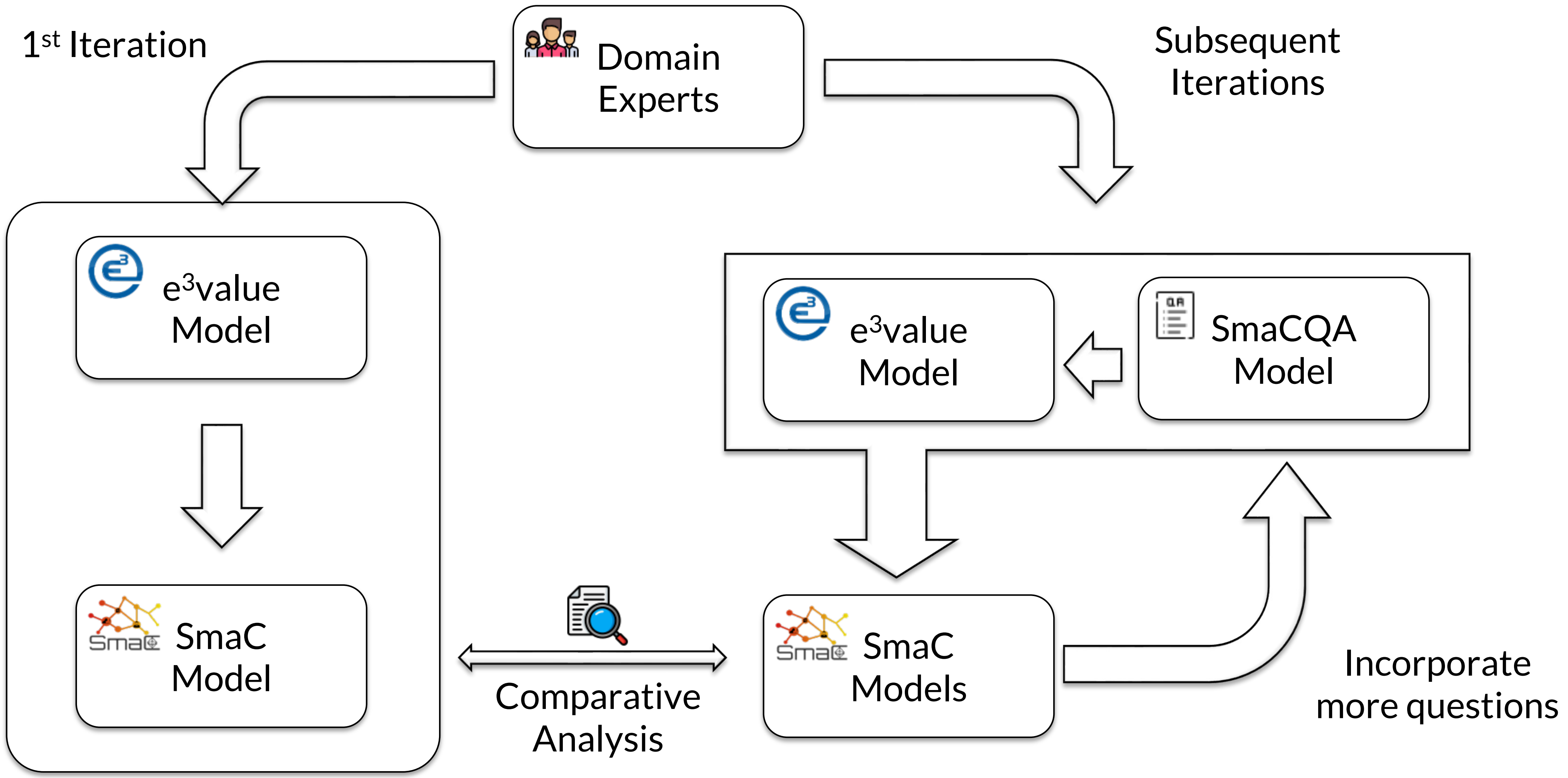


- Compiler version
- Contract's name
- Inclusion of more information on actors
- Function's name, modifiers,
- Events

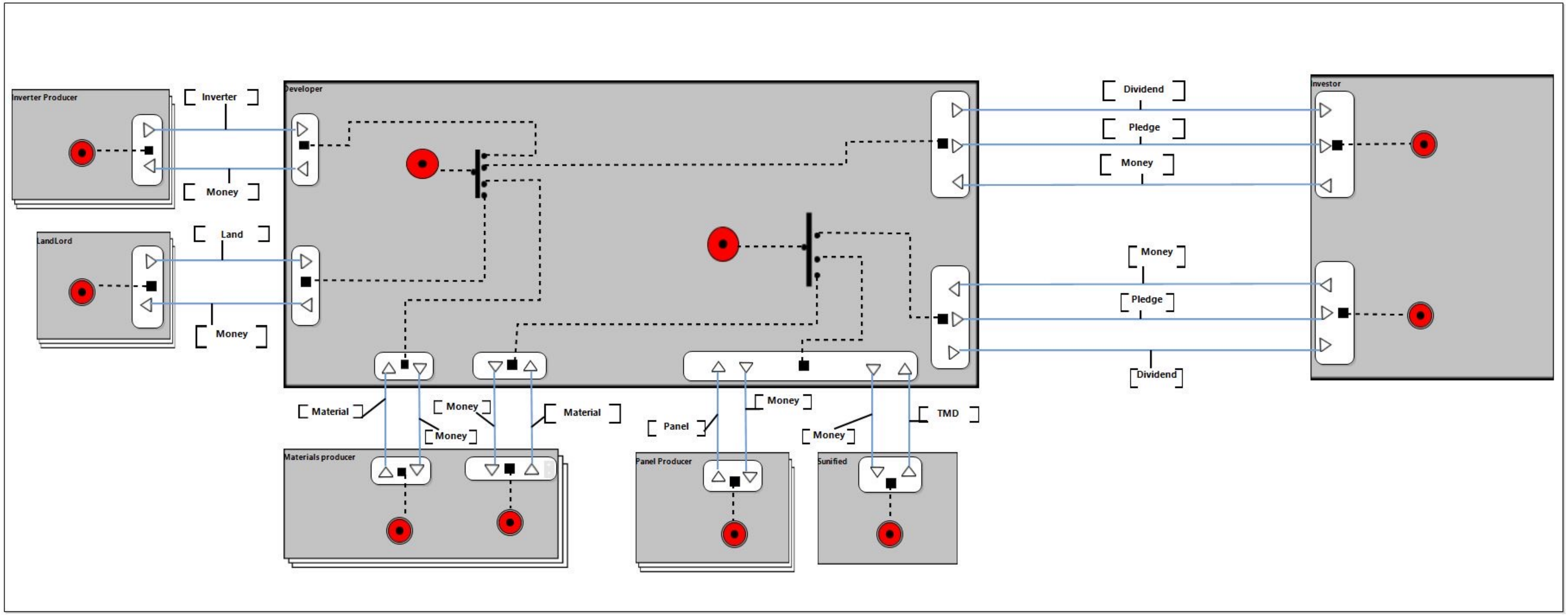

Compiler Version
Insert Smart Contract Compiler Version
Compiler Version
Tooltip about define version
Aceptar

Agenda

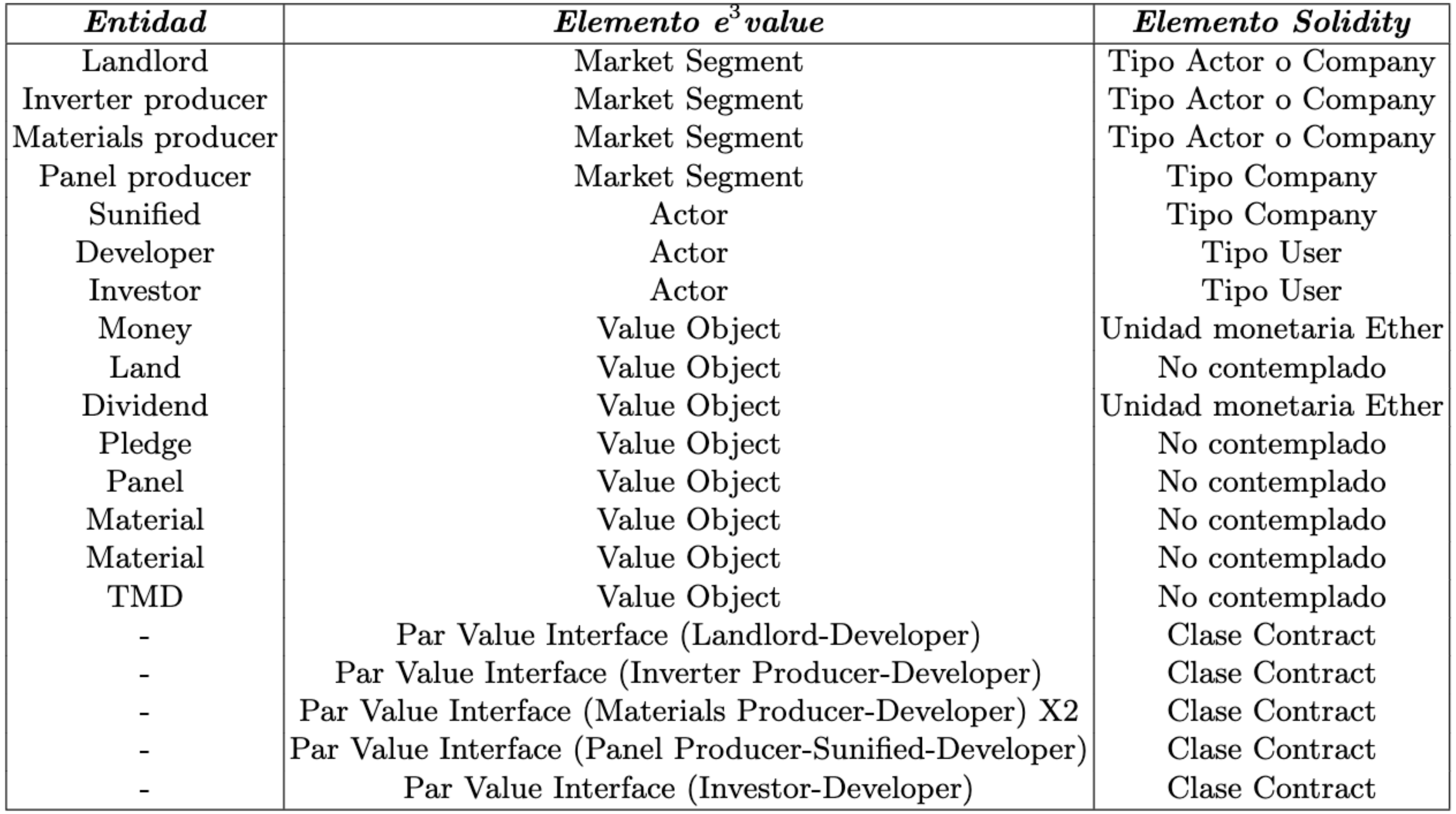
Evaluation Protocol



Business Model – Solar Farm



Business Model – Solidity concepts correspondence



Entidad	*Elemento e^3value*	*Elemento Solidity*
Landlord	Market Segment	Tipo Actor o Company
Inverter producer	Market Segment	Tipo Actor o Company
Materials producer	Market Segment	Tipo Actor o Company
Panel producer	Market Segment	Tipo Company
Sunified	Actor	Tipo Company
Developer	Actor	Tipo User
Investor	Actor	Tipo User
Money	Value Object	Unidad monetaria Ether
Land	Value Object	No contemplado
Dividend	Value Object	Unidad monetaria Ether
Pledge	Value Object	No contemplado
Panel	Value Object	No contemplado
Material	Value Object	No contemplado
Material	Value Object	No contemplado
TMD	Value Object	No contemplado
-	Par Value Interface (Landlord-Developer)	Clase Contract
-	Par Value Interface (Inverter Producer-Developer)	Clase Contract
-	Par Value Interface (Materials Producer-Developer) X2	Clase Contract
-	Par Value Interface (Panel Producer-Sunified-Developer)	Clase Contract
-	Par Value Interface (Investor-Developer)	Clase Contract

Results & Main findings

Code generation improvements

Iteration	*#SmaCQA Questions (Total)*	*#LOC Solidity (Total)*	*SmaCQA* **Model Elements**
1	0 (No SmaCQA)	178	smart contract(s), structs, variables & functions
2	18	397	variables, modifiers & functions
3	35	518	Struct data type
4	42	540 + 327 (Smart Contract token)	Token (Smart Contract), functions & variables

IT.1
- Generate struct type for actors (identify actors involved)
- Generate smart contracts (identify pair value interfaces)
- Generate functions (identify value exchanges)

IT.2
- Generation of a tax property (Tax Legal Question 2.2)
- Generation of a tax actor collector (Tax Collector Legal Question 2.2.1)
- Generate of modifiers for change the tax amount & tax collector (Derived from 2.2 & 2.2.1)

IT.3
- Generation of a legal age property (Minimun Age Legal Question 2.1)
- Generation of a struct type for value objects (Value Object Tangible Question 1.5)
- Generation of a modifier for verify the legal age (Derived from 2.1)

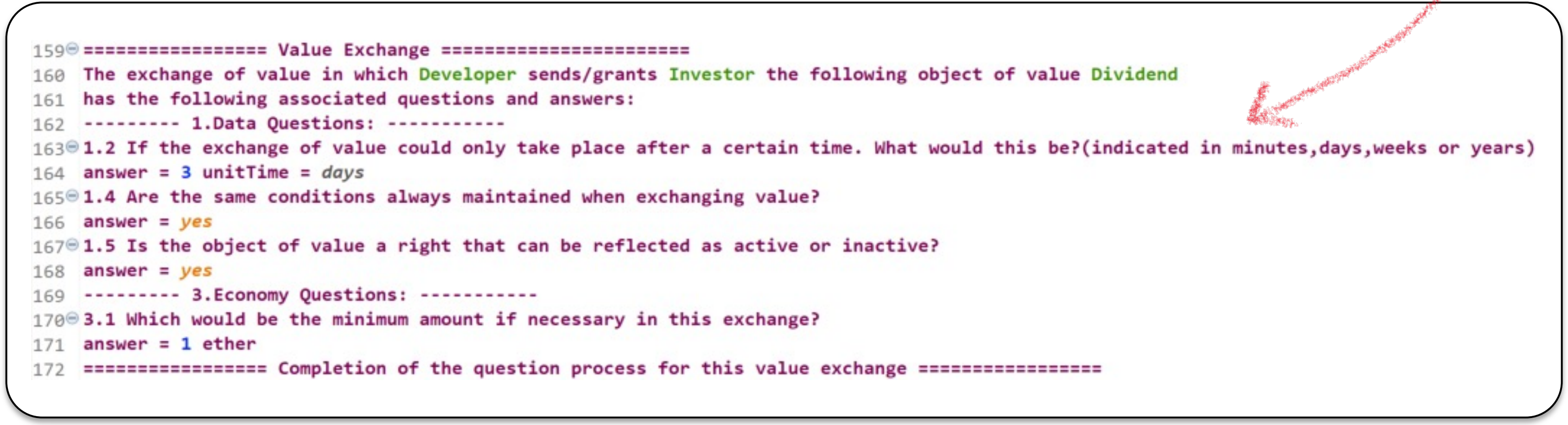
IT.4
- Generate time property (Data Question 1.2)
- Generate boolean property to represent a right (Value Object Right Question 1.5)
- ERC-20 smart contract (Value Object Fungible Token Question 1.5)
- Generate property to represent the minimum monetary amount (Economy Question 3.1)
- Generation of a modifier for time control of the value exchange (Derived from 1.2)
- Generation of a function to modify the time interval (Derived from 1.2)
- Generation of a modifier to control if the actor has the minimum balance amount (Derived from 3.1)

Conclusions

- Limitations of e^{3}value
- SmaCQA increases completiveness
- Complete fungible token model
- SmaCQA to be tested in different domains

A closer look

Value Exchange must be completed in 3 days

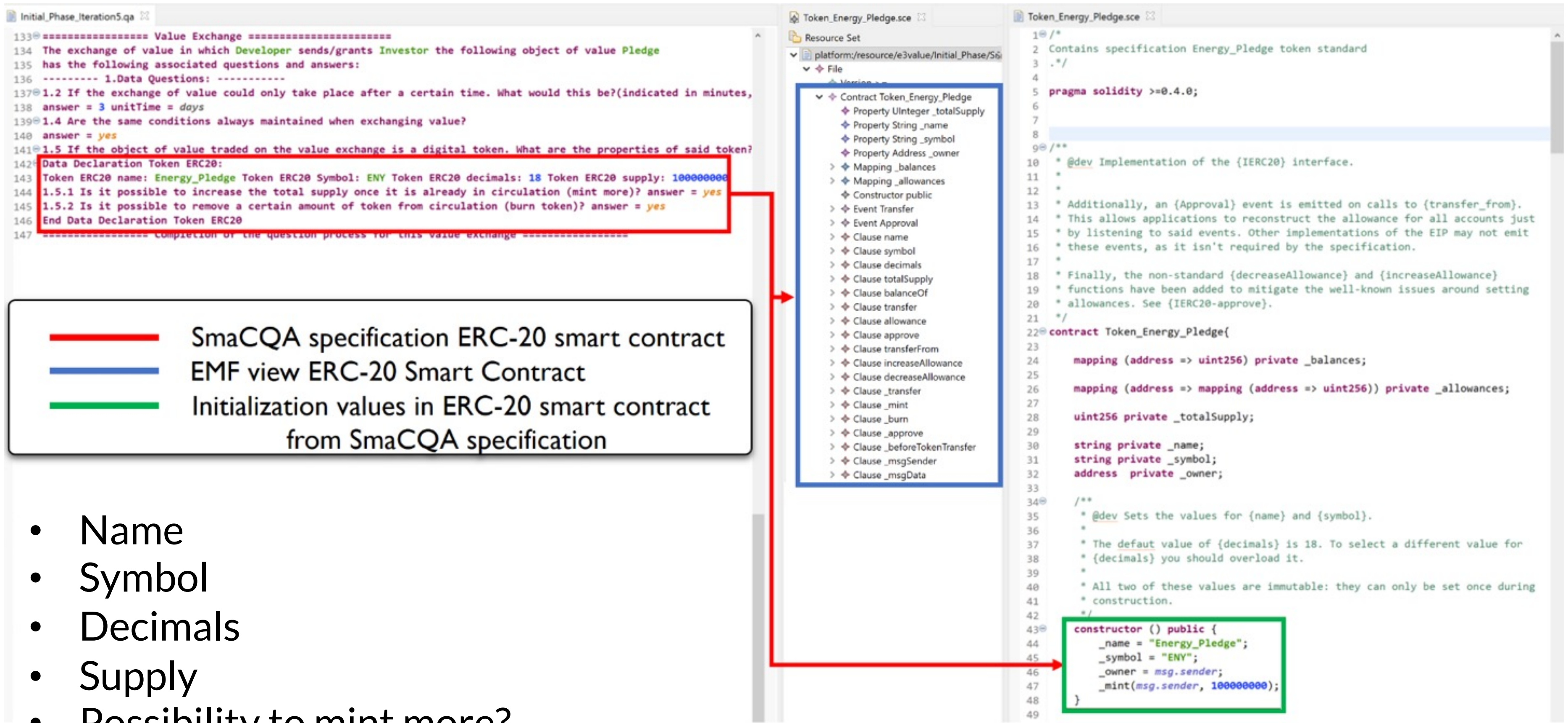


```
================= Value Exchange =======================
The exchange of value in which Developer sends/grants Investor the following object of value Dividend
has the following associated questions and answers:
--------- 1.Data Questions: -----------
1.2 If the exchange of value could only take place after a certain time. What would this be?(indicated in minutes,days,weeks or years)
answer = 3 unitTime = days
1.4 Are the same conditions always maintained when exchanging value?
answer = yes
1.5 Is the object of value a right that can be reflected as active or inactive?
answer = yes
--------- 3.Economy Questions: -----------
3.1 Which would be the minimum amount if necessary in this exchange?
answer = 1 ether
================= Completion of the question process for this value exchange =================
```

Model gathers some more data regarding different element types (before they are to be used=, such as modifiers, numeric variables, etc.

A closer look

ERC-20 Specification



Initial_Phase_Iteration5.qa

```
================= Value Exchange =======================
The exchange of value in which Developer sends/grants Investor the following object of value Pledge
has the following associated questions and answers:
--------- 1.Data Questions: -----------
1.2 If the exchange of value could only take place after a certain time. What would this be?(indicated in minutes,
answer = 3 unitTime = days
1.4 Are the same conditions always maintained when exchanging value?
answer = yes
1.5 If the object of value traded on the value exchange is a digital token. What are the properties of said token?
Data Declaration Token ERC20:
Token ERC20 name: Energy_Pledge Token ERC20 Symbol: ENY Token ERC20 decimals: 18 Token ERC20 supply: 100000000
1.5.1 Is it possible to increase the total supply once it is already in circulation (mint more)? answer = yes
1.5.2 Is it possible to remove a certain amount of token from circulation (burn token)? answer = yes
End Data Declaration Token ERC20
================= Completion of the question process for this value exchange ================
```

Token_Energy_Pledge.sce

```
Resource Set
  platform:/resource/e3value/Initial_Phase/S6
    File
      Version
      Contract Token_Energy_Pledge
        Property UInteger _totalSupply
        Property String _name
        Property String _symbol
        Property Address _owner
        Mapping _balances
        Mapping _allowances
        Constructor public
        Event Transfer
        Event Approval
        Clause name
        Clause symbol
        Clause decimals
        Clause totalSupply
        Clause balanceOf
        Clause transfer
        Clause allowance
        Clause approve
        Clause transferFrom
        Clause increaseAllowance
        Clause decreaseAllowance
        Clause _transfer
        Clause _mint
        Clause _burn
        Clause _approve
        Clause _beforeTokenTransfer
        Clause _msgSender
        Clause _msgData
```

Token_Energy_Pledge.sce

```
/*
Contains specification Energy_Pledge token standard
.*/

pragma solidity >=0.4.0;



/**
 * @dev Implementation of the {IERC20} interface.
 *
 *
 * Additionally, an {Approval} event is emitted on calls to {transfer_from}.
 * This allows applications to reconstruct the allowance for all accounts just
 * by listening to said events. Other implementations of the EIP may not emit
 * these events, as it isn't required by the specification.
 *
 * Finally, the non-standard {decreaseAllowance} and {increaseAllowance}
 * functions have been added to mitigate the well-known issues around setting
 * allowances. See {IERC20-approve}.
 */
contract Token_Energy_Pledge{

    mapping (address => uint256) private _balances;

    mapping (address => mapping (address => uint256)) private _allowances;

    uint256 private _totalSupply;

    string private _name;
    string private _symbol;
    address  private _owner;

    /**
     * @dev Sets the values for {name} and {symbol}.
     *
     * The defaut value of {decimals} is 18. To select a different value for
     * {decimals} you should overload it.
     *
     * All two of these values are immutable: they can only be set once during
     * construction.
     */
    constructor () public {
        _name = "Energy_Pledge";
        _symbol = "ENY";
        _owner = msg.sender;
        _mint(msg.sender, 100000000);
    }

```

🟥 SmaCQA specification ERC-20 smart contract
🟦 EMF view ERC-20 Smart Contract
🟩 Initialization values in ERC-20 smart contract from SmaCQA specification

- Name
- Symbol
- Decimals
- Supply
- Possibility to mint more?
- Possibility to burn specify amount?

Agenda

Recap

Blockchain networks providing a computational platform

- Trust-less | Immutability | Transparency
- Disambiguation + Disintermediation

Smart Contracts as the way to explode such infrastructure

- IT – Strategy gap
- Essential + Accidental Complexity ➪ Tooling needed

Models to the rescue

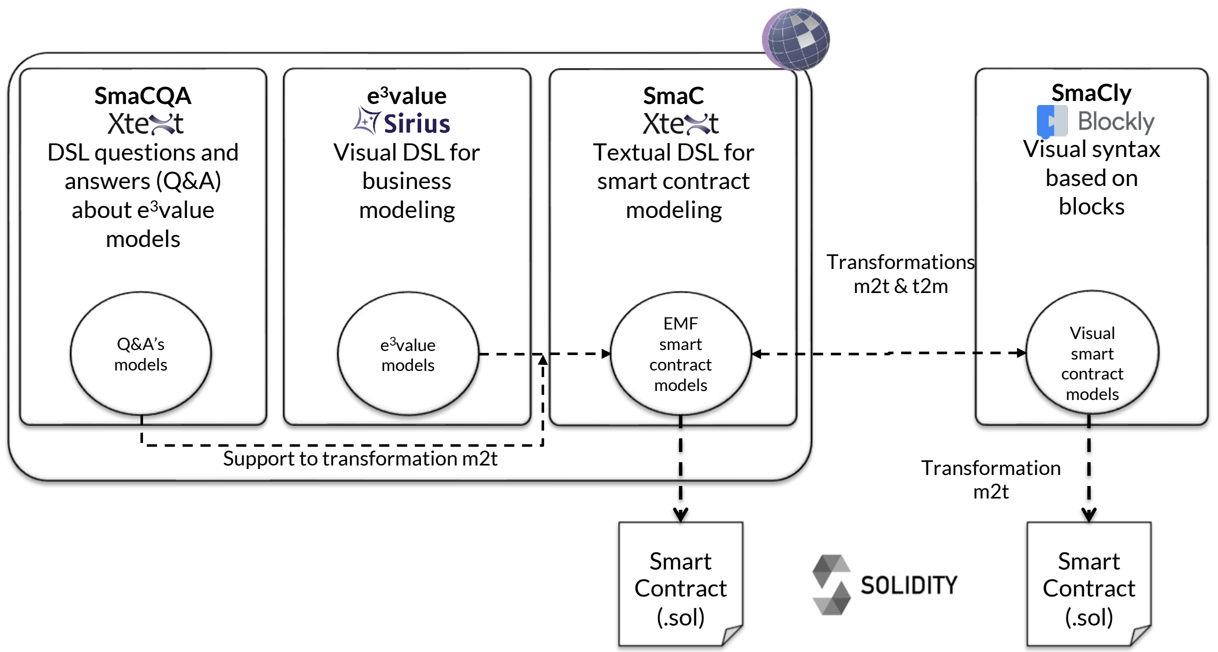
Raise the level of abstraction at which Smart Contracts are developed /designed

SMART CONTRACTS DEMOCRATIZATION

Recap

Building a model-based toolkit for the development of smart contracts

- SmaC – a textual DSL to bring contracts to the realm of models
- SmaCly – a visual DSL to enable graphical development of contracts
  - Code completion, contextual assist, development pattern, syntax and semantic checking, etc. … for each DSL
  - Integration
- Mappings to shorten the distance with domain experts
- SmaCQA – a textual DSL to improve contract generation from business models


SmaCQA
Xtext
DSL questions and answers (Q&A) about e³value models
Q&A's models
e³value
Sirius
Visual DSL for business modeling
e³value models
SmaC
Xtext
Textual DSL for smart contract modeling
EMF smart contract models
Support to transformation m2t
Transformations m2t & t2m
SmaCly
Blockly
Visual syntax based on blocks
Visual smart contract models
Transformation m2t
Smart Contract (.sol)
SOLIDITY
Smart Contract (.sol)

Road ahead – Steps been taken

SmaC

- SmaC2e^{3}value mapping
- Automatic deployment of modelled contracts
- Move SmaC to the Web (SmaCly is already a Web IDE)

SmaCly

- CEP-based mining of developers work

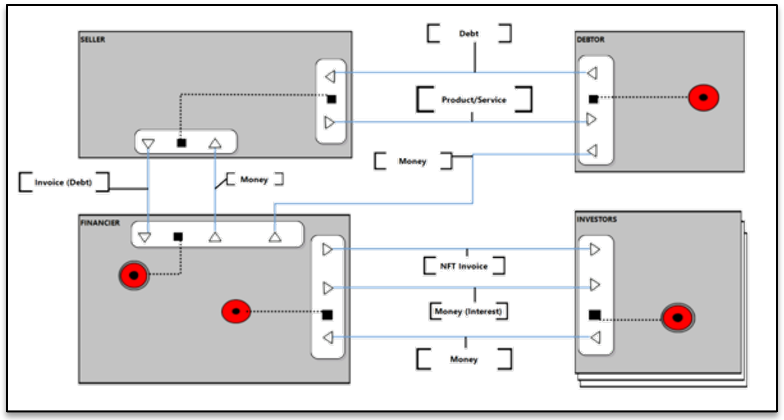
SmaCQA

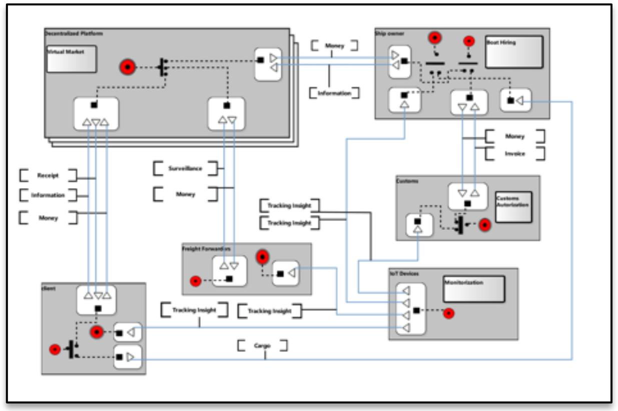
- Visual syntax
- Test SmaCQA in different domains to expand the set of supported questions

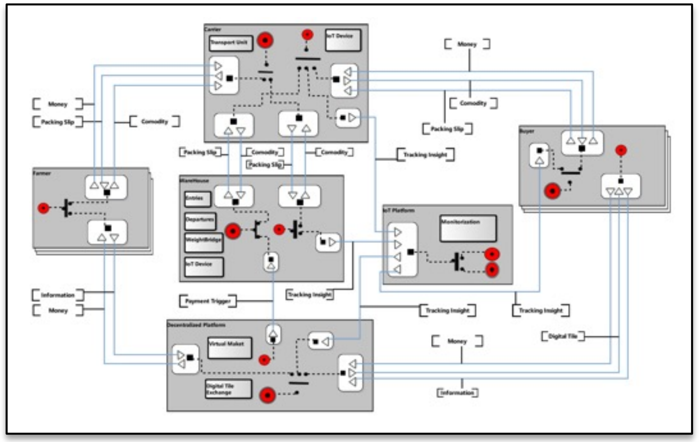
Road ahead – Steps been taken

SmaCQA

- Test SmaCQA in different domains to expand the set of supported questions
  - Factoring (investment mechanism)
  - Supply Chain







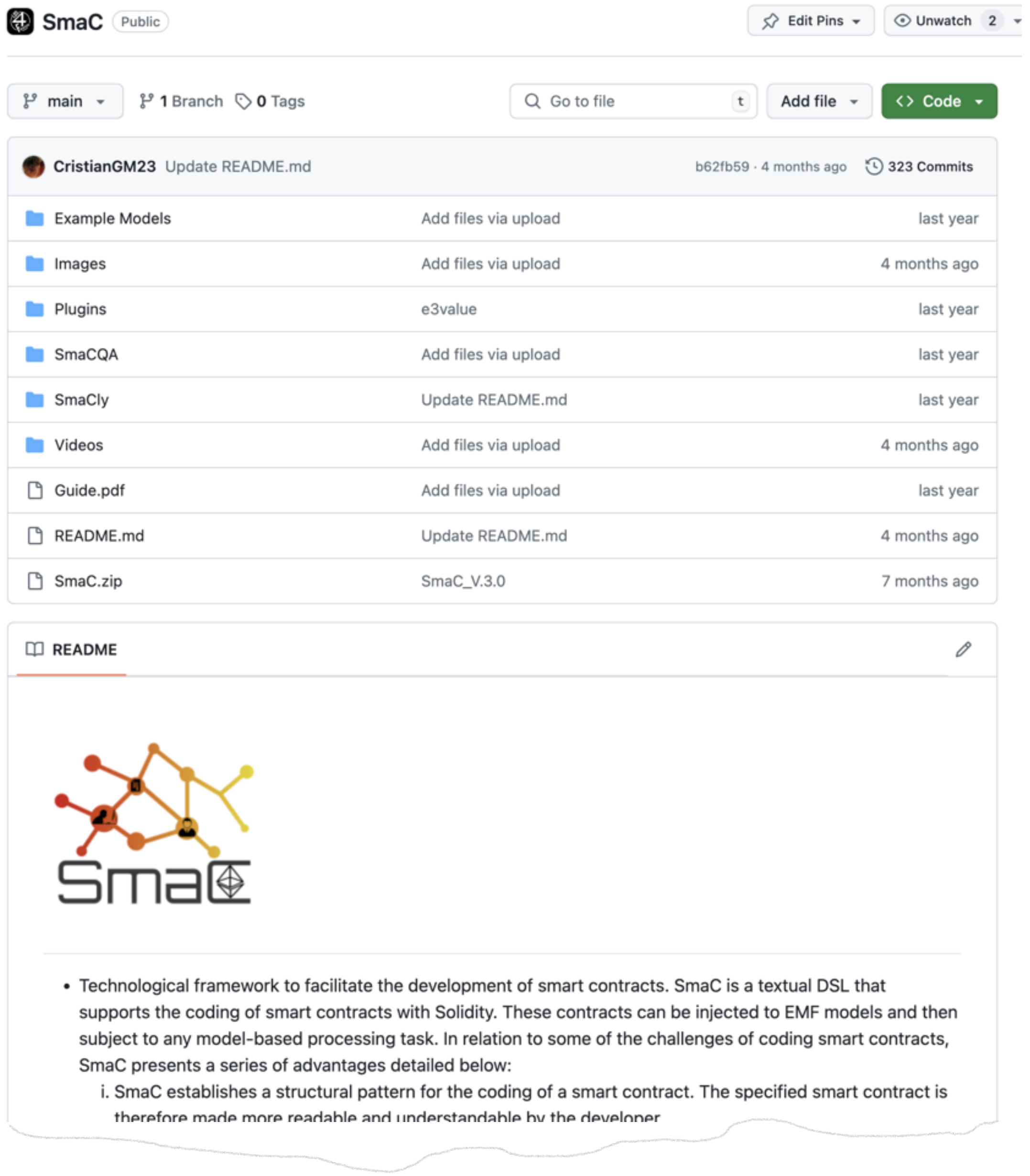
Former SmaCQA	
#Categories	3
#Questions	23

Current SmaCQA	
#Categories	6
#Questions	75

Visit our *playground*

SmaC repo on GitHub

`https://github.com/The4Fantastics/SmaC`



SmaC Public

Edit Pins · Unwatch 2

main · 1 Branch · 0 Tags · Go to file · Add file · Code

CristianGM23 Update README.md · b62fb59 · 4 months ago · 323 Commits

Example Models	Add files via upload	last year
Images	Add files via upload	4 months ago
Plugins	e3value	last year
SmaCQA	Add files via upload	last year
SmaCly	Update README.md	last year
Videos	Add files via upload	4 months ago
Guide.pdf	Add files via upload	last year
README.md	Update README.md	4 months ago
SmaC.zip	SmaC_V.3.0	7 months ago

README

SmaC

- Technological framework to facilitate the development of smart contracts. SmaC is a textual DSL that supports the coding of smart contracts with Solidity. These contracts can be injected to EMF models and then subject to any model-based processing task. In relation to some of the challenges of coding smart contracts, SmaC presents a series of advantages detailed below:
   i. SmaC establishes a structural pattern for the coding of a smart contract. The specified smart contract is therefore made more readable and understandable by the developer

CommIT
Computer Science & Information Technologies

Universidad Rey Juan Carlos

SmaCQA: de modelos de negocio a contratos inteligentes

URJC Escuela Técnica Superior de Ingeniería Informática

Juan Manuel Vara [Juancho]

juanmanuel.vara@urjc.es

@jmvara