FDTF DLT WG Call Notes

*11 July 2017*

# Attendees

* Mike Bennett
* Rob Nehmer
* Pinar Emirdag
* Pete Rivett
* Inessa Collier

# What we covered

* Updates
* Proof of Concept
* Programming languages
* Links from conceptual model / ontology to the code
* Actions on this

# Updates

## Ontology for requirements

No new updates.

## Identifier requirements

 - conversation with Olivier on ISO 307 - DT

 - several SGs e.g. SG1 on Reference Data - most active and relevant.

 - not sure how to join?

## OMG general liaison with ISO

 - follow up with Richard Soley.

# Proof of Concept

* Push ahead with the PoC
* have something to demonstrate to ISO TC307.

## Programming Languages

Standard programming language (de facto or otherwise) Can we get someone who has the programming skill?

* Not likely to be the case. There are different DLTs which use their own languages e.g. Solidity on Ethereum. Bitcoin BC has its own. New ones use functional programming languages.
* So it will not be realistic to expect any one language to be imposed on these. Very platform specific.

That reinforces the argument for a model driven approach - can generate into different programming languages.

### Links to the code

Identified a few candidate languages.

See if the ones we thought about are still relevant.

 - Dig up the notes on those

 - find people who are familiar with those.

Is it realistic to use familiar languages as an analog?

PR - makes more sense to show how different paradigms can be related to the CIM.

 - e.g. the functional programming languages

 - LISP / Haskell / Clojure

Derive both structural and procedural aspects of the proposed solution (SmartContract etc.)

Hopefully structural will be the same regardless.

Structural as RDF and OWL? Would simplify what we demonstrate.

How to then show model driven development of at least 2 very different kinds of program language content.

How? Are there plug-ins for MDA for any of these, for UML tools??

What about e.g. Java generation? - Too primitive for what we want to show here.

## Next steps to make this happen?

### Development Tools

For examples of functional languages. But the ones listed above are not the specific ones being used in the existing DLTs.

Not look at the above ones, but the ones people are actually using.

### Actions

Follow up with Inessa on specifics.

FLs work on Immutability.

Could do a functional programming style even in Javascript see e.g. immutable.js as a library for that.

Can we have one logical design as above, for which the program specifics would be physical manifestations?

### Things to address on this:

 - what should appear in the SmartLedger?

 - what should the record be?

Having worked out what the record should look like

From that, identify what would make up the Smart Contract for each of these.

Thereby effectively arrive at the code incrementally.

# NEXT STEPS

* Start an email chain before next week
* Establish set of candidate languages to look at
* Add links from process ontology to IR Swaps contract ontology (MB)