Dconf 2019 Keynote

Laeeth Isharc
Symmetry Investments

Alternative Investment Management

Adoption of an emerging language at a $4.8bn hedge fund

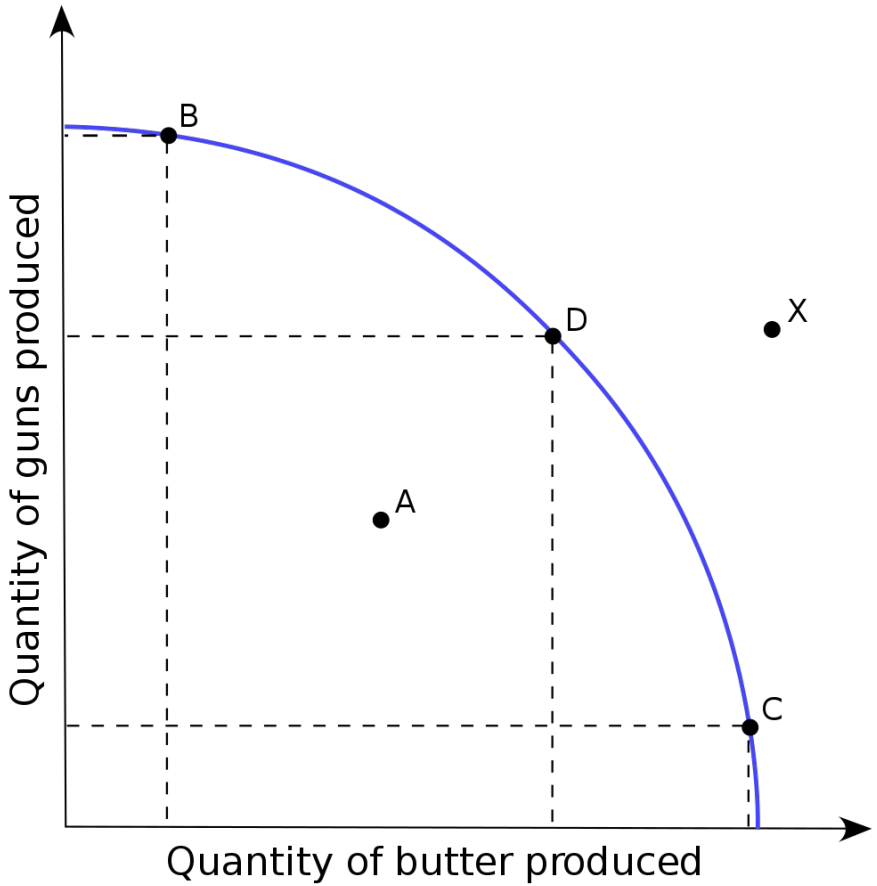
Alternative Investment Management

- Symmetry Investments is an alternative investment management company.
- Alternative means that if everybody does the same thing and you understand why, but you can see a better way then in the alternative domain you have a choice.

Symmetry Win-Win Principle

- Dialectical reconciliation of seeming-opposites.
- The need to compromise presumes you are already at the optimum and therefore must choose.
- Refuse to accept premature compromise
- Nature has many secrets yet to discover – have the courage to be open to trying

Efficient Frontier



Premature Compromises?

- Productivity or performance (Python vs C)
- Expressiveness or readability
- Modeling power or plasticity
- Adaptability to change alongst planned dimensions versus brittleness in the face of change you did not expect

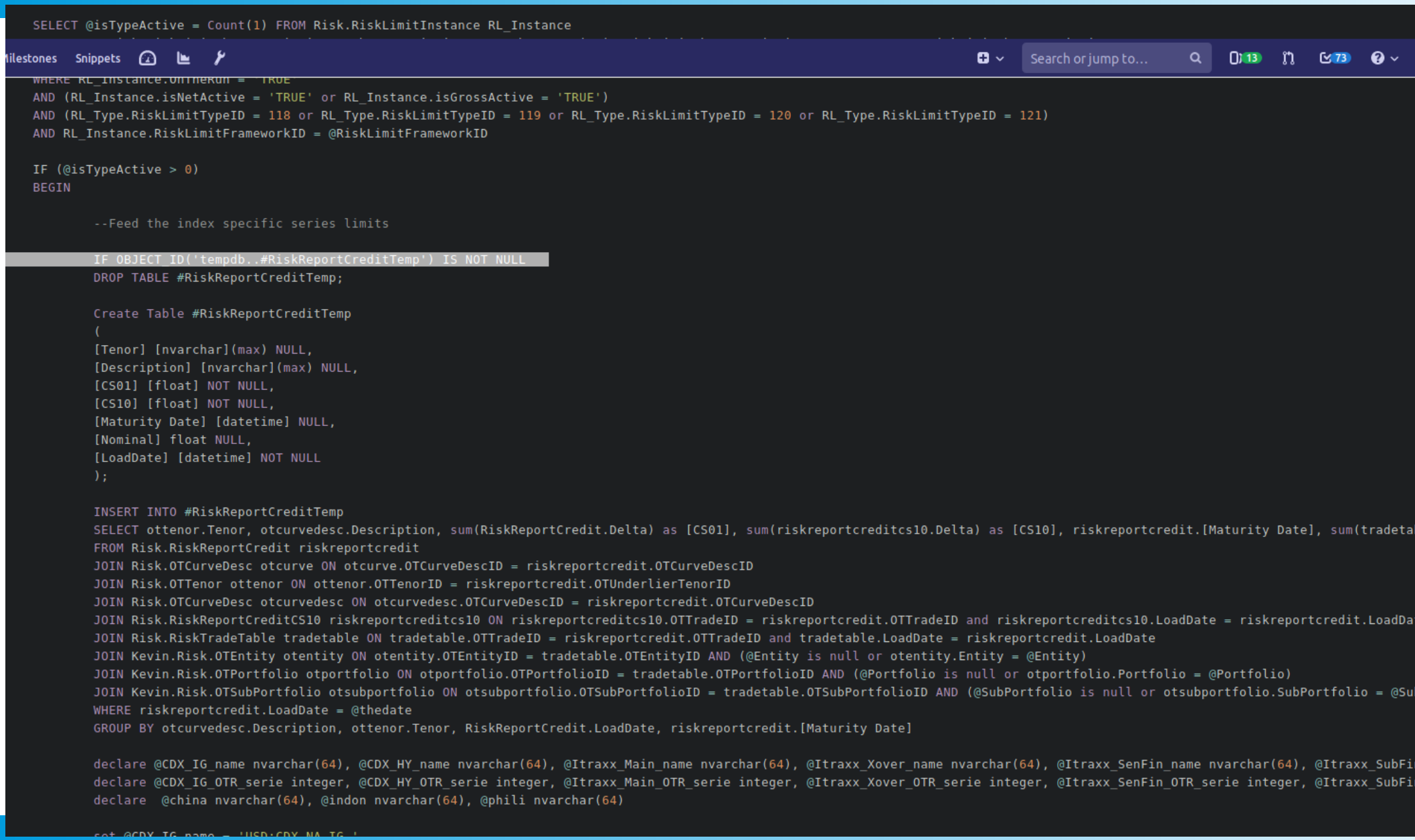
Cost and Choice

- Theoretical goal of the firm: maximise profits
- Reality for large publicly-owned companies? Something else.
- Risk aversion – but relevant dimension is social - unconventional choices seen as riskier.
- Life is risk. Social risk isn't really real.
- Creates opportunities for others operating according to different principles and context

Selected Symmetry Ownership Principles

- What would I do if I owned the business?
- Now, considering I don’t, how can we address any unavoidable social factors.
- It’s okay to be unconventional if you’re also serious.
- Failure is okay, but consider magnitude of the downside.

!


SELECT @isTypeActive = Count(1) FROM Risk.RiskLimitInstance RL_Instance
Milestones
Snippets
Search or jump to...
13
73
WHERE RL_Instance.OnTheRun = 'TRUE'
AND (RL_Instance.isNetActive = 'TRUE' or RL_Instance.isGrossActive = 'TRUE')
AND (RL_Type.RiskLimitTypeID = 118 or RL_Type.RiskLimitTypeID = 119 or RL_Type.RiskLimitTypeID = 120 or RL_Type.RiskLimitTypeID = 121)
AND RL_Instance.RiskLimitFrameworkID = @RiskLimitFrameworkID
IF (@isTypeActive > 0)
BEGIN
--Feed the index specific series limits
IF OBJECT_ID('tempdb..#RiskReportCreditTemp') IS NOT NULL
DROP TABLE #RiskReportCreditTemp;
Create Table #RiskReportCreditTemp
(
[Tenor] [nvarchar](max) NULL,
[Description] [nvarchar](max) NULL,
[CS01] [float] NOT NULL,
[CS10] [float] NOT NULL,
[Maturity Date] [datetime] NULL,
[Nominal] float NULL,
[LoadDate] [datetime] NOT NULL
);
INSERT INTO #RiskReportCreditTemp
SELECT ottenor.Tenor, otcurvedesc.Description, sum(RiskReportCredit.Delta) as [CS01], sum(riskreportcreditcs10.Delta) as [CS10], riskreportcredit.[Maturity Date], sum(tradeta
FROM Risk.RiskReportCredit riskreportcredit
JOIN Risk.OTCurveDesc otcurve ON otcurve.OTCurveDescID = riskreportcredit.OTCurveDescID
JOIN Risk.OTTenor ottenor ON ottenor.OTTenorID = riskreportcredit.OTUnderlierTenorID
JOIN Risk.OTCurveDesc otcurvedesc ON otcurvedesc.OTCurveDescID = riskreportcredit.OTCurveDescID
JOIN Risk.RiskReportCreditCS10 riskreportcreditcs10 ON riskreportcreditcs10.OTTradeID = riskreportcredit.OTTradeID and riskreportcreditcs10.LoadDate = riskreportcredit.LoadDa
JOIN Risk.RiskTradeTable tradetable ON tradetable.OTTradeID = riskreportcredit.OTTradeID and tradetable.LoadDate = riskreportcredit.LoadDate
JOIN Kevin.Risk.OTEntity otentity ON otentity.OTEntityID = tradetable.OTEntityID AND (@Entity is null or otentity.Entity = @Entity)
JOIN Kevin.Risk.OTPortfolio otportfolio ON otportfolio.OTPortfolioID = tradetable.OTPortfolioID AND (@Portfolio is null or otportfolio.Portfolio = @Portfolio)
JOIN Kevin.Risk.OTSubPortfolio otsubportfolio ON otsubportfolio.OTSubPortfolioID = tradetable.OTSubPortfolioID AND (@SubPortfolio is null or otsubportfolio.SubPortfolio = @Su
WHERE riskreportcredit.LoadDate = @thedate
GROUP BY otcurvedesc.Description, ottenor.Tenor, RiskReportCredit.LoadDate, riskreportcredit.[Maturity Date]
declare @CDX_IG_name nvarchar(64), @CDX_HY_name nvarchar(64), @Itraxx_Main_name nvarchar(64), @Itraxx_Xover_name nvarchar(64), @Itraxx_SenFin_name nvarchar(64), @Itraxx_SubFi
declare @CDX_IG_OTR_serie integer, @CDX_HY_OTR_serie integer, @Itraxx_Main_OTR_serie integer, @Itraxx_Xover_OTR_serie integer, @Itraxx_SenFin_OTR_serie integer, @Itraxx_SubFi
declare @china nvarchar(64), @indon nvarchar(64), @phili nvarchar(64)

Symmetry Integration Language

- Think many times before writing your own language
- Are you sure you want to write a standard library too?
- Purpose: domain-driven design. Shared attention from practitioners and technologists towards common representation of code and data.

SIL Standard Library

Good artists copy, great artists steal.

- Steve Jobs

SIL Standard Library

Good artists copy, great artists steal.

- Steve Jobs

=> Let’s steal Phobos

Stealing Phobos

```
void registerHandlersMath(ref Handlers handlers)
{

  import std.mathspecial;
  handlers.openModule("math");
  scope(exit) handlers.closeModule();

  static foreach(f;AliasSeq!(   fabs,         sqrt,         sin,          cos,          tan,          asin,
                                acos,         atan,         sinh,         cosh,         tanh,         asinh
                                acosh,        atanh,        /*log*/       log2,         log10,        logb,
                                log1p,        exp,          exp2,         expm1,        ceil,         floor,
                                round,        lround,       trunc,        rint,         lrint,        nearbyint,
                                rndtol,       gamma,        logGamma,     sgnGamma,     digamma,
logmdigamma,
                                logmdigammaInverse,         erf,          erfc,         normalDistribution,
                                normalDistributionInverse,  atan2,        fmod,         remainder,    isIdentical,
                                fmax,         fmin,         gammaIncomplete,  ))

      handlers.registerHandler!f;

}
```

SIL: registerHandler

```
void registerHandler(string name, Function handler, SILdoc doc = SILdoc.init, string file = __FILE__, int line = __LIN
{
        warnOverwrite(name, file, line);
        requestHandlers[name] = Hdlr(Variable(handler), file, line, doc);
}

void registerHandler(alias handler)(string name = __traits(identifier, handler), string file = __FILE__, int line = __
{
        this.registerHandler(name, createFunction!handler(name), getSILdoc!handler, file, line);
}
```



```
void registerConstructorHandler(alias handler,ArgumentNames...)(string name = __traits(identifier, handler), string fi
{
        this.registerHandler(name, createFunction!(handler,ArgumentNames)(name), getSILdoc!handler, file, line);
}

void aliasHandler(string newName, string oldName, string file = __FILE__, int line = __LINE__)
{
        warnOverwrite(newName, file, line);
        requestHandlers[newName] = requestHandlers[oldName];
}

// make struct/class type T storable in a Variable and make its members accessible
void registerType(T)(string name=T.stringof, string file = __FILE__, int line = __LINE__)
{
```

Symmetry Integration Language

- 4,552 SloC for language
- 5,998 SloC for standard library
- 131 uses of static if
- 35 static foreach
- 117 string mixins
- 112 __traits

SIL code sample

```
// Bucketing
sumThisBucket = { (tbl, keyBuckets, keyValues, bucket) => (tbl |> filterTableByCol( keyBuckets, {entry => entry == bucket }))[keyValues] |> sum }
bucketEntries = { (tbl, keyBuckets, keyValues, arrBuckets) => arrBuckets |> mapa( {entry => [entry, tbl |> sumThisBucket(keyBuckets, keyValues, entry)] } ) }

// Balance sheet functions
balShtUSD                     = { bs => [bs.endQuote, bs.endNominal, bs.endFXPrice, bs.endAccrual] \
                                                |> zip \
                                                |> map(array) \
                                                |> map({ args => chooseIf(args[0] < 20.0, (args[1] * args[2]), (args[1] * args[0] / 100.0 + args[3]) * args[2
                                    }

balShtNative                  = { bs => [bs.endQuote, bs.endNominal, bs.endFXPrice, bs.endAccrual] \
                                          |> zip \
                                          |> map(array) \
                                          |> map({ args => chooseIf(args[0] < 20.0, (args[1]), (args[1] * args[0] / 100.0 + args[3])) }) \
                                  }

netBalSht                     = {entries          => entries |> balShtUSD |> sum}
irDelta                       = {entries          => entries.usdIrDelta |> sum}
netBalShtNativeCcy            = {entries          => entries |> balShtNative |> sum}
uniqueISINs                   = {entries          => entries.isin |> sort |> uniq |> array}
grossBalShtSingleISIN         = {(entries, isin)  => entries |> filterISIN(isin) |> netBalSht |> abs }
grossBalShtSingleISINNativeCcy = {(entries, isin)  => entries |> filterISIN(isin) |> netBalShtNativeCcy |> abs }
grossBalSht                   = {entries          => entries |> uniqueISINs |> map({ x => entries |> grossBalShtSingleISIN( x )}) |> sum}
grossBalShtNativeCcy          = {entries          => entries |> uniqueISINs |> map({ x => entries |> grossBalShtSingleISINNativeCcy( x )}) |> sum}

// Limits Functions
filterLimitType               = { (bs, typeIn)           => bs |> filterTableByCol("type", {lt => lt == typeIn} ) }
filterLimitCharacteristic     = { (bs, characteristicIn) => bs |> filterTableByCol("characteristic", {cc => cc == characteristicIn} ) }
filterLimits                  = { (limitsTable, entityIn, subPortfolioIn, typeIn, characteristicIn) => limitsTable \
                                                |> filterEntity( entityIn ) \
                                                |> filterSubPortfolio( subPortfolioIn ) \
                                                |> filterLimitType( typeIn ) \
                                                |> filterLimitCharacteristic( characteristicIn ) \
                                  }
```

Symmetry Integration Language

Sneaky Purpose – cultural transformation

1) Version control
2) Code review
3) Testing
4) Declarative Programming
5) Integration between practitioners and programmers / transformation of cultural values

Secrets

We've been taught that truth is conventional, but it's simply something that people agree on. What's important is to discover a new truth."

All great businesses are based on secrets.

– Peter Thiel

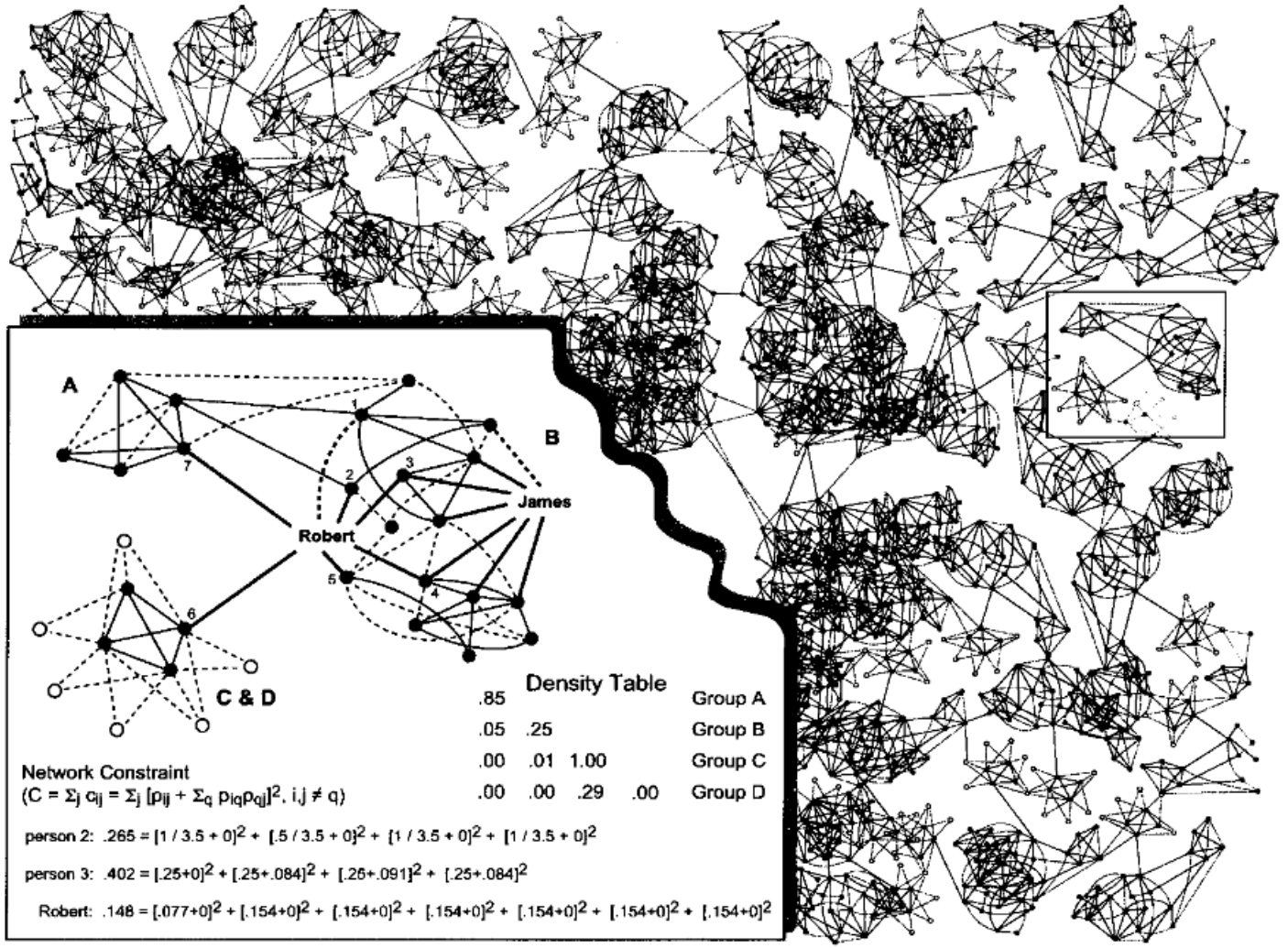
The best secrets are hidden in plain sight

- Tom Demark

Structural Hole Theory

- Structural holes exist in social networks when there is a lack of a direct contact or tie between two or more entities (Burt, 1992).
- How to benefit from competition in social networks and their intersecting relationships ?

Small-world Networks



Structural Hole Theory and Good Ideas

Behaviour, opinion, and information are more homogeneous within than between groups.

People focus on activities inside their own group, creating holes in the information flow between groups.

Robert is better positioned than James for the social capital of brokerage.

Import/Export Business in Ideas

- People who bridge structural holes have an advantage in detecting and developing rewarding opportunities. **Information arbitrage is their advantage.**
- They are able to see early, see more broadly, and translate information across groups.
- Brokerage across the structural holes between groups provides **a vision of options otherwise unseen**.

D and Structural Holes

Organizations using the D Language

AdRoll

Marketing Platform

Integral parts of AdRoll's business are built on D.

♥ *D is for Data Science*

ArabiaWeather

Leading weather services provider in the Arab world

❝ Only D drawback: it's so pleasant that coding in any other language becomes a nuisance. ❞

D TechTalk (Arabic)

AREX

Real-time exchange for B2B credit

❝ D is a superb language for high volume, low latency order matching. ❞

Github Hiring

Auburn Sounds

Audio processing

❝ Nothing quite matches D's power. ❞

Github ♥ Testimonial

CERERIS

Hardware and Software development

❝ We use D to develop MVC web framework for embedded systems. ❞

Github

Cut Through Recordings

Digital Signal Processing

❝ If you are looking for a powerful language that can support many programming styles, compiles natively, and is blazing fast, then D may be worth checking out. ❞

Github ♥ Testimonial

eBay

One of the world's largest marketplaces

Large scale data mining tools.
Command line tools in D

Github

ecratum

Supplier Management tool for SME

Core applications (Public API, Support app) use D.

Hiring

Emsi

Data-driven modelling

N-dimensional dataset processing, in-memory data manipulation

DConf talk Github

It’s okay to be unreasonable

When you grow up you tend to get told the world is the way it is and you're life is just to live your life inside the world.

Life can be much broader once you discover one simple fact: **Everything around you that you call life was made up by people that were no smarter than you and you can change it, you can influence it, you can build your own things that other people can use.**

Once you learn that, you'll never be the same again.

– Steve Jobs

Imagination/Courage/Beginnings

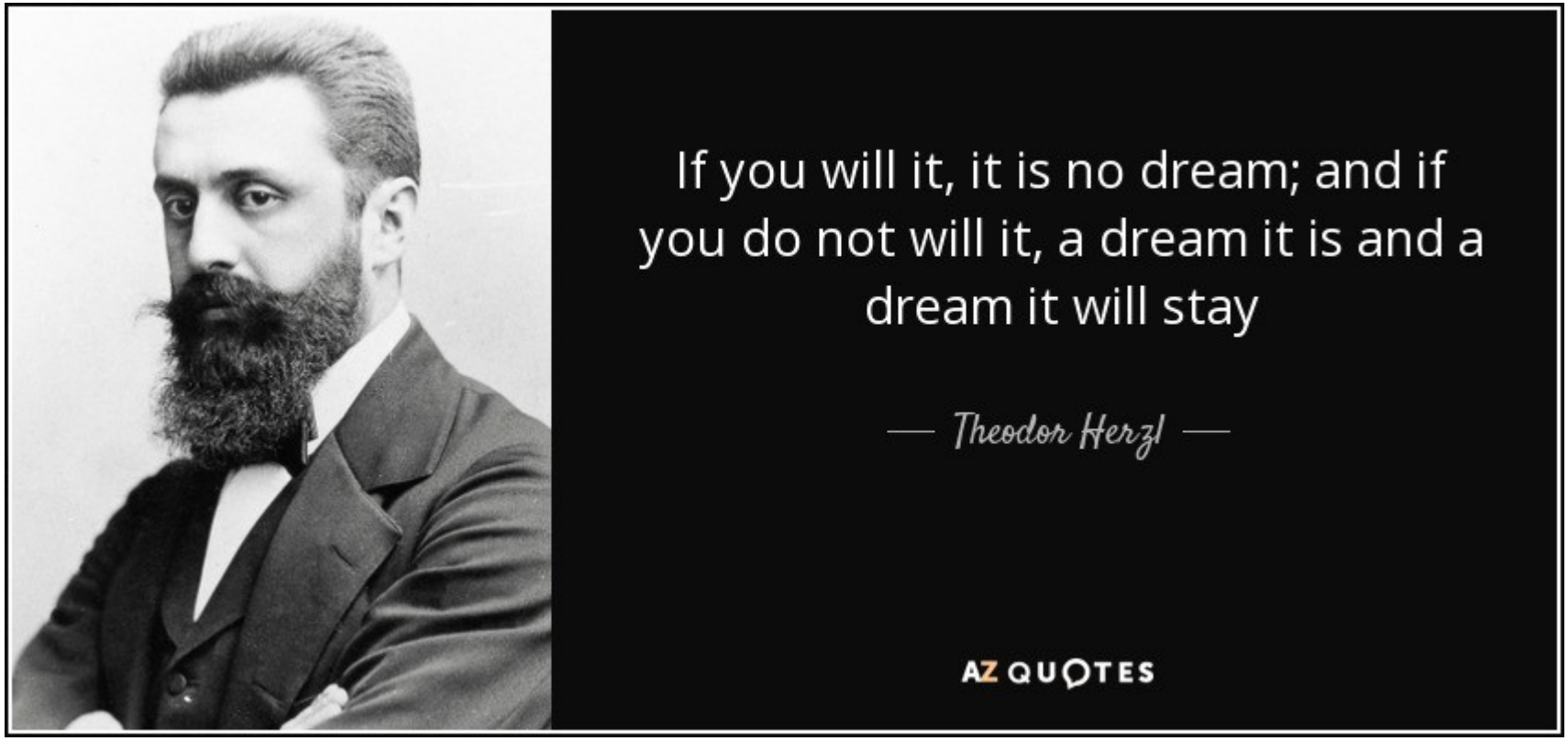
Until one is committed, there is hesitancy, the chance to draw back, always ineffectiveness. [T]he moment one definitely commits oneself, then providence moves too.

All sorts of things occur to help one that would never otherwise have occurred. raising in one's favour all manner of unforeseen incidents, meetings and material assistance which no man could have dreamed would have come his way.

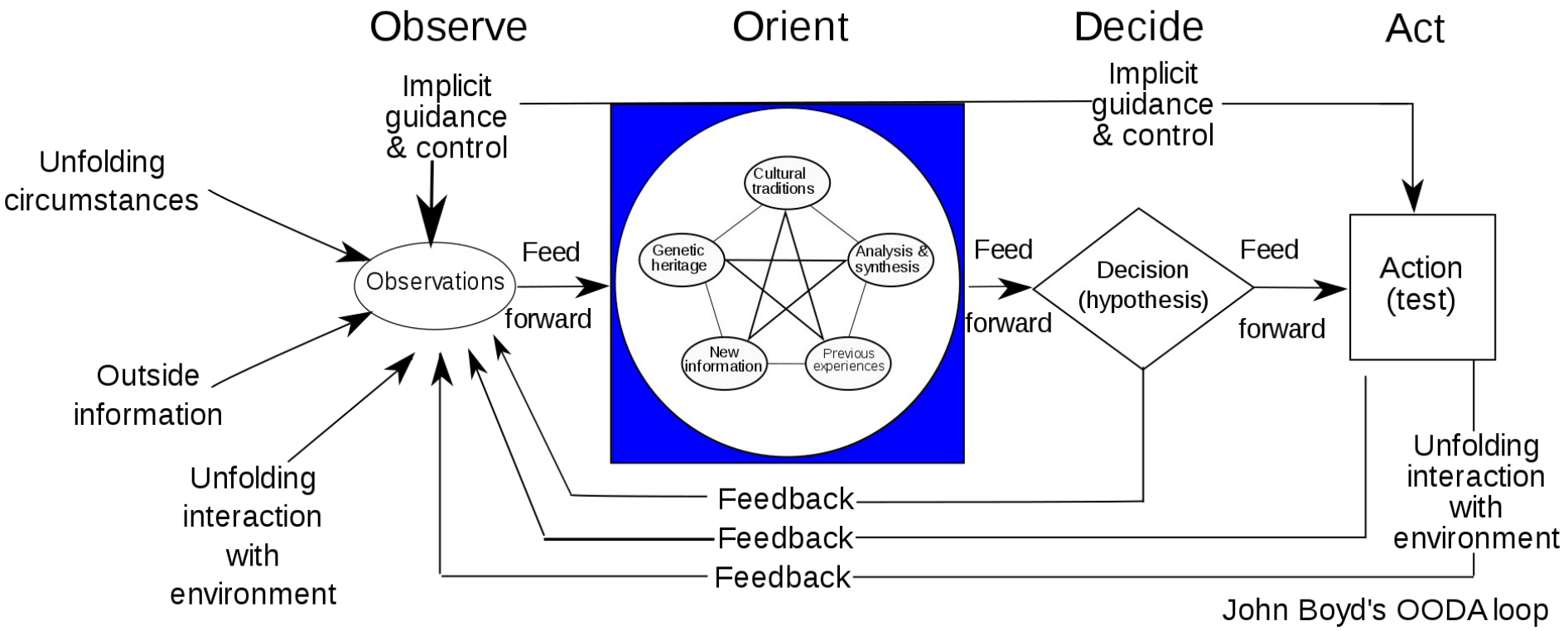
Whatever you can do or dream you can, begin it. Boldness has genius, power and magic in it. Begin it now.”

– William Hutchison Murray

Beginnings


If you will it, it is no dream; and if you do not will it, a dream it is and a dream it will stay
— Theodor Herzl —
AZ QUOTES

Plasticity for Suits



Unexpected Benefits

1. Courageousness/ Span of control/ resourcefulness
2. Information Arbitrage
3. Talent
4. Marshalling / type conversion for glue
5. Plasticity = adaptiveness and strategic optionality

We’re Hiring Capable People

For D / F# /Typescript in

- London
- Hong Kong
- Singapore

Laeeth Isharc laeeth@kaleidic.io

https://symmetryinvestments.com/careers

References

Burt – Structural Holes and Good Ideas

Israel Kirzner on the nature of entrepreneurship

James Buchanan – Cost and Choice (mises.org)

William Boyd