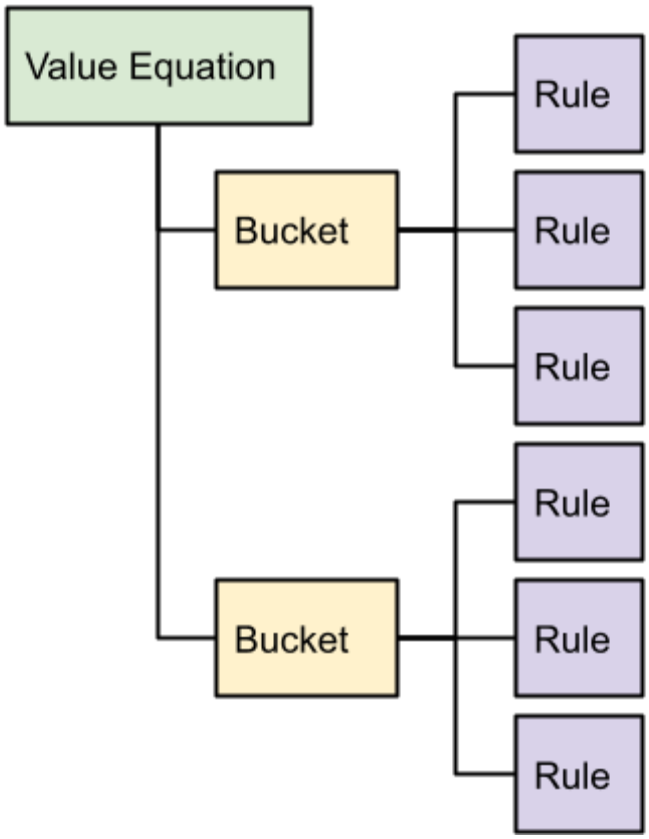
Value Equation Structure



An example of some work contributed to several deliverables:
Another example of materials purchased for several deliverables:
Value flow example

Definitions

Value equation - The value equation prescribes how revenue is distributed to ~~all participants (active affiliates)~~ contributors in a context (which could be a project, or larger than a project). In other words, it is used to compute a % of total revenue for ~~every participant/contributor~~ the contributors who are selected by the value equation, taking into consideration different forms of contributions and different modulators [see more on OVN wiki]
Bucket - A bucket is a division of the value equation that is given a percentage of the revenue to be distributed. It also defines a filter, which determines which contributions go into that bucket. Buckets are sequenced.
Rule - A rule defines the treatment of a type of contribution: both the equation used to compute its value, and whether it will be treated as debt (eligible for distributions until paid off), equity (eligible for distributions forever), or one-time (eligible for only one distribution).
A rule can also specify secondary filters to refine the selection of contributions for which the rule applies.

Percentage Behavior:

- Remaining Percentage takes the percent of the remaining distribution amount after each bucket is processed. The last bucket should be 100%. This option distributes only as much as is defined by the claims affected by the bucket. There may be a leftover amount, which will become part of the next bucket in the sequence.
- Straight Percentage divides up the total into the defined percentages. The bucket percentages should add up to 100%. This option distributes the entire amount included in the bucket percentage of the total. Recipients may receive more than their claim amount. There will be no leftover.

Treatment of different event types:

- Consumption events are not contributions, although money and work that went into the consumables (for example, polymers for 3D printing) are contributions.
    - The amount of a contribution that went into a process is determined by the quantity of the consumable that was consumed, and each contribution's share of the value of the consumable resource.
- Use events are not contributions, although money and work that went into the usable resource (for example, a community-funded 3D printer) are contributions.
    - The amount of a contribution that went into a process is determined by the use value of the use event (hours * value per unit of use), and each contribution's share of the value of the usable resource.
- Citation events are not contributions, although money and work that went into the citable resource are contributions.

    - The citation event may be valued by percentage of the value of the other inputs to a process, or by quantity of the common unit of the value equation (for example, Canadian Dollars). That value is used to compute the amounts of the contributions to the cited resource.

- Production events may be contributions if a value equation bucket gives credit for production. (Sensorica has not done so, but other networks want to.)

- Work events are direct contributions.

Examples:


PV Characterization M1
Change
Add a Bucket
Make This Live
Context: PV characterization
Bucket percentages are calculated using the method: Straight percentage
PV Characterization Milestone 1, with bucket percentages based on work done
Bucket 1 - 15.16% - Coordination and Facilitation
Change
Delete
Add a Bucket Rule
Filter method: Process
Rule for: Time Contribution
Change
Delete
Test Rule
Claim calculation: Until paid off, quantity * 20
Filter by:
Bucket 2 - 40.21% - R&D
Change
Delete
Add a Bucket Rule
Filter method: Process
Rule for: Time Contribution
Change
Delete
Test Rule
Claim calculation: Until paid off, quantity * 30
Filter by:
Bucket 3 - 19.35% - Outreach
Change
Delete
Add a Bucket Rule
Filter method: Process
Rule for: Time Contribution
Change
Delete
Test Rule
Claim calculation: Until paid off, quantity * 25
Filter by:
Bucket 4 - 25.27% - Documentation
Change
Delete
Add a Bucket Rule
Filter method: Process
Rule for: Time Contribution
Change
Delete
Test Rule
Claim calculation: Until paid off, quantity * 25
Filter by:

Process PV test Change Add a Bucket

Context: **PV characterization**

Make This Test

Bucket percentages are calculated using the method: **Straight percentage**

Can this simpler VE match the one using Tibi's percentages?

Bucket 1 - 100% - Doing the work Change Delete Add a Bucket Rule

Filter method: **Process**

Rule for: **Time Contribution** Change Delete

Claim calculation: **Until paid off, quantity * 20**

Filter by: **Coordination and Facilitation**

Test Rule

Rule for: **Time Contribution** Change Delete

Claim calculation: **Until paid off, quantity * 30**

Filter by: **Design considerations**

Test Rule

Rule for: **Time Contribution** Change Delete

Claim calculation: **Until paid off, quantity * 25**

Filter by: **Mapping interest**

Test Rule

Rule for: **Time Contribution** Change Delete

Claim calculation: **Until paid off, quantity * 25**

Filter by: **Outreach for resources**

Test Rule

3D course Change | Add a Bucket | Make This Live

Context: **Course 3D**

Bucket percentages are calculated using the method: **Remaining percentage**

Distribution of revenue from the 3D course

Bucket 1 - 5% - for SENSORICA infrastructure Change | Delete

Distribute directly to: **SENSORICA**

Bucket 2 - 10% - Initial Financial Contributions Change | Delete | Add a Bucket Rule

Filter method: **Order**

Rule for: **Payment** Change | Delete | Test Rule

Claim calculation: **Until paid off, quantity * 1.2**

Filter by:

Bucket 3 - 20% - Initiation Change | Delete | Add a Bucket Rule

Filter method: **Order**

Rule for: **Time Contribution** Change | Delete | Test Rule

Claim calculation: **Until paid off, quantity * 24**

Filter by:

Bucket 4 - 100% - Rolling cost Change | Delete | Add a Bucket Rule

Filter method: **Order**

Rule for: **Receipt** Change | Delete | Test Rule

Claim calculation: **Until paid off, value**

Filter by:

Rule for: **Resource use** Change | Delete | Test Rule

Claim calculation: **One distribution, quantity * 5**

Filter by:

Bucket 5 - 100% - Rolling Roles Change | Delete | Add a Bucket Rule

Filter method: **Order**

Rule for: **Time Contribution** Change | Delete | Test Rule

Claim calculation: **One distribution, quantity * 20**

Filter by:

Bucket 6 - 100% - Support Roles Change | Delete | Add a Bucket Rule

Filter method: **Order**

Rule for: **Time Contribution** Change | Delete | Test Rule

Claim calculation: **Until paid off, quantity * 20**

Filter by:

Algorithm

The Value Equation Tutorial is more informative, but here's some simplified

pseudo-code:

For each bucket:
    Give the bucket its percentage of the amount_to_be_distributed
        (see explanation of *straight* and *remaining* percentages above)
    If the bucket amount goes directly to one agent:
        give it to that agent
    else:
        run the value equation for that bucket:
            gather the contributions that match the bucket filter:
- date range filter is pretty simple, gets all contributions for the date range for the context agent (this is intended mostly for non-production related contributions)
- Order, delivery or process filters gathers all of the contributions that are inputs to the selected object (order, delivery or process), including tracing back through all previous processes that fed into the selected object.
    - In all of those cases, compute the amount of the contribution that went into the selected object.
        - (see examples on next pages)

            For each contribution:
- get the bucket rule that applies
    - (if no rule applies, that contribution is out of this run)
- compute the value of the contribution using the bucket rule *claim creation equation*
- if the contribution has a claim already, use that claim
    - (if the claim is zero (has been paid off), then that contribution is out of this run)

  else create a claim using the full value of the contribution
- pro-rate the distribution amount for this contribution according to the share of the contribution that applied to this bucket
- adjust the claim with the pro-rated amount according to the bucket rule claim treatment:
    - until paid off (debt-like), or
    - forever (equity-like), or
    - one distribution (don't give it any more)

Now all the contributions that apply to this bucket have been collected, and the amount of each contribution that applies to this bucket has been computed.
So we allocate the amount to be distributed to this bucket to the claims proportionately:

1) **Straight percentage:** the amount to be distributed divided by the claim's percentage of the total claims for that bucket, or: amt / (total_claims / claim). So, for example, if the amount to be distributed is $100, and the total value of all the claims for that bucket are 4, and the value of each claim is 2, then each claim gets $50: 100 / (4/2) = 50. If each of the claims was for 200, they would still just get $50. The amount of the bucket is distributed, no more and no less.
2) **Remaining percentage:** For the example above, each claim gets $2. For an example where each claim was 80, each claim gets $50. In other words, in remaining, you get your claim, but it is capped by the bucket % amount.
    a) When you select remaining percentage, all of the claims in all of the buckets might not use up the total amount to be distributed to all buckets. As stated above, the percentage for the last bucket should be 100%, and then it will allocate the remainder. Otherwise, you might get some left over, and then need to figure out how to distribute that.

In other words, each bucket runs its own value equation, for its own contributions, using the percentage of the distribution amount given to that bucket.

Math:

The claim creation equation in a bucket rule allows mathematical expressions that can use any mathematical operator, parentheses to manage precedence, and any of these variables from the Economic Event that the claim will be based on:

- quantity
- value
- valuePerUnit
- valuePerUnitOfUse (of the Resource used)
- pricePerUnit (of the Resource Type)

More variables could be added, but those are the ones that can be used as of this writing.

The various filters select the Economic Events and their associated claims to be included in a bucket or bucket rule.

In straight percentage, the values of the selected claims are summed, and the total amount to be distributed by the bucket is allocated to each claim based on its percentage of the total claims for the bucket.

See description above for remaining percentage.

Most of the above could be done in a spreadsheet, and can be described by mathematical notation, although we have seen, both often and recently, that spreadsheets can be buggy and difficult to understand, too.

The other mathematical operation is graph traversal, following the resource flows that went into an order or shipment, and traversing back into consumption, use and citation events to gather the contributions that went into them. The graph traversals involved are pretty specialized, and I don't know how you would describe them in declarative mathematical notation, nor do them in a spreadsheet without extreme convolutions. They could *possibly* be declared in an advanced matrix manipulation language like APL, but it is notoriously a write-only language.

Contexts:

If a graph traversal crosses a context agent boundary (for example, from PV characterization to Digital Fab), and the other context does not use the same value equation✦, the PV characterization value equation stops and gives the distribution amount to Digital Fab, to be re-distributed by their value equation.

✦ An example where two contexts might use the same value equation is where both were projects in Sensorica, and both were using a Sensorica value equation.

Spreadsheet vs NRP Value Equation

Spreadsheet:

- Pluses:
  - Familiarity: many people know how to use them
  - Some degree of transparency: all of the code is in the spreadsheet
- Minuses:
  - Can get so complicated that they are buggy, with no debug tools
    - Easy to make mistakes
  - Once they get that complicated, they are no longer transparent
    - You may not be able to understand them anymore yourself
    - For example, https://groups.google.com/forum/#!topic/sensorica/1w6eeft3BDI
  - They can't traverse graphs without extreme skills, and even then, not to the degree required for value equations
  - Their code is not open source, can't add features easily
    - But: plugins! A lucrative market!
  - Requires exporting data from NRP or providing other spreadsheets where people enter data (that's if you want to abandon NRP). And the selections and export are not in the spreadsheet: that is, are not transparent.

NRP Value Equation:

- Pluses:
  - More structured than a spreadsheet
    - In a spreadsheet, you need to do a lot of structuring yourself
    - In other words, once you know how they work, NRP VEs may be easier and less buggy
  - The basic structure of the value equation may be more clear (or not, depend on the VE)
  - Different people can do different value equations and compare in sandbox
    - Programmers can create comparison tools as required
    - A "Copy VE" feature could be added to make it easier
  - A complicated equation in a spreadsheet *can* be less complicated in NRP
  - Graph traversal built in.
  - Code is open source, we can add features
  - And the selections of events are transparent (on the face of the value equation).
- Minuses:
  - Not as familiar, has a learning curve
  - The outer structure of the value equation may be more clear than a spreadsheet, but the inner details of traversals and pro-rating distribution amounts are buried in the code.
    - You can actually read the code, but it's probably not as familiar as

        spreadsheet functions.
        - But, see Bayle's log proposal
    - Graph traversals are advanced math, not for the faint of heart
        - But necessary anyway, can't be avoided

More about graph traversals:

- http://www.mkbergman.com/1020/the-age-of-the-graph/
- graph theory for kids
- http://facebook.github.io/graphql/
- http://www.slideshare.net/slidarko/the-gremlin-traversal-language
- http://neo4j.com/blog/open-cypher-sql-for-graphs/

More examples

An example of some work contributed to several deliverables:

- If I contribute $40 worth of work… to a process that created 4 deliverables… then I added $10 of value to each deliverable.
- If 2 of them were sold on an order, I added $20 of value to the order.
- When do I get my other $20? When the other 2 are sold. Maybe. Might depend on the other buckets in the Value Equation.

Another example of materials purchased for several deliverables:

- If I buy 100 meters of optical fiber for $200, the value per unit is $2.
- If a process uses 15 meters, and produces 4 deliverables, that's $7.50 of optical fiber per unit.
- If one of those 4 deliverables was consumed in creating a purchased product, that's $1.875 per purchased product.
- Might be awhile till I get my $200 back, but I am a patient person. And the value equation says I will get 20% extra for my patience. So that's $2.25 from this distribution.

Value flow example

Incoming Value Flows for Joint-type transducer: Test134

Total accumulated value per unit : $260.00

1 **Resource:** Joint-type transducer: Test134 Value per unit: 260.00
Explanation: Value per unit is composed of the value of the inputs on the next level:
2 **Process:** Assemble joint-type transducer to test manufacturing recipe starting 2014-10-28 ending 2014-10-28 Production qty: 1
2 **Time Contribution:** *2014-11-02 from Lynn Foster to Joint-type transducer 2 HR R&d mechanics* Value per unit: 20
*Explanation: Value per parent unit: 40 = Resource Type value per unit: 20 * Event quantity: 2 / Process production qty: 1*
2 **Time Contribution:** *2014-10-25 from Bob Haugen to Joint-type transducer 3 HR Manufacturing - mechanics* Value per unit: 20
*Explanation: Value per parent unit: 60 = Resource Type value per unit: 20 * Event quantity: 3 / Process production qty: 1*
2 **Resource Consumption:** 2014-10-25 from Joint-type transducer to Joint-type transducer 1 EA Mosquito glass capillary: Glass capillaries from VWR Value per unit: 100.00
*Explanation: Value per parent unit: 100.00 = Resource value per unit: 100.00 * Event quantity: 1 / Process production qty: 1*
3 **Resource:** Mosquito glass capillary: Glass capillaries from VWR Value per unit: 100.00
Explanation: Value per unit is composed of the value of the inputs on the next level:
4 **Receipt:** 2014-11-03 from VWR to SENSORICA 1 EA Mosquito glass capillary: Glass capillaries from VWR Value per unit: 100
*Explanation: Value: 100 = Resource value per unit: 100 * Event quantity: 1*
2 **Resource Consumption:** 2014-10-25 from Joint-type transducer to Joint-type transducer 1 EA Mosquito joint: 156
3 **Resource:** Mosquito joint: 156
2 **Resource Consumption:** 2014-10-25 from Joint-type transducer to Joint-type transducer 1 EA Optical connector part: Mating Sleeve ST to ST
3 **Resource:** Optical connector part: Mating Sleeve ST to ST
2 **Resource Consumption:** 2014-10-25 from Joint-type transducer to Joint-type transducer 1 EA Mosquito lever: Test134 Value per unit: 60.00
*Explanation: Value per parent unit: 60.00 = Resource value per unit: 60.00 * Event quantity: 1 / Process production qty: 1*
3 **Resource:** Mosquito lever: Test134 Value per unit: 60.00
Explanation: Value per unit is composed of the value of the inputs on the next level:
4 **Process:** make levers to test manufacturing recipe starting 2014-10-27 ending 2014-10-27 Production qty: 1
4 **Resource Consumption:** 2014-10-25 from Optical fiber coating to Optical fiber coating 1 EA Mirror on fiber: nice mirror Value per unit: 20.00
*Explanation: Value per parent unit: 20.00 = Resource value per unit: 20.00 * Event quantity: 1 / Process production qty: 1*
5 **Resource:** Mirror on fiber: nice mirror Value per unit: 20.00
Explanation: Value per unit is composed of the value of the inputs on the next level:
6 **Process:** make mirror and Computer program product to test manufacturing recipe starting 2014-10-26 ending 2014-10-26 Production qty: 4
6 **Time Contribution:** *2014-10-25 from Bob Haugen to Optical fiber coating 2 HR Manufacturing - optics* Value per unit: 20
*Explanation: Value per parent unit: 10 = Resource Type value per unit: 20 * Event quantity: 2 / Process production qty: 4*
6 **Citation:** 2014-10-25 from Optical fiber coating to Optical fiber coating Joint-type transducer design: Optical Design - joint-capilaire_fibre-optique Serge Value per unit: 80.00
Explanation: Value: 1 = Event quantity: 1