Building a Useful Goal Statement - Raphael L. Vitalo, Ph.D.

There is a minimal set of ideas required to document the knowledge someone needs to perform a task successfully.[1] These are:

- the goal of the task,
- the resources needed prior to starting it (inputs),
- the results the task must produce (outputs),
- the method the performer uses to transform inputs into outputs (process),
- the tests done to ensure successful execution of the task (feedback), and
- a list of the individuals or groups with whom to communicate while performing the task (coordination), the information to be exchanged, and the protocols to follow with each.[2]

This article provides you with guidance for representing the first component of this set of knowledge, the task's goal.

Goal

A goal describes what it is the task must accomplish. It is the reference point a performer needs to guide his or her decision making and action taking. "Good" decisions and actions produce the results specified in the goal and satisfy its conditions for task performance. "Bad" decisions and actions do not. A goal also enables you to test logically the integrity of the remaining knowledge. For example, one way to test whether the statement of outputs is correct is to ask whether the specified outputs accomplish the goal. If they do not, then the statement of outputs is invalid or incomplete, assuming the goal is correct. Another test of the knowledge's integrity is whether the method described for doing the task can be performed without violating the conditions specified in the goal. If it cannot, the method is not valid.

Components of a Complete Goal Statement

Each useful goal statement includes six components: a "to" statement that tells the summary result to be produced, a "for" statement that tells who is to benefit from accomplishing the goal, a "by" statement that names the task to be implemented to produce the result, a "so that" statement that lists the benefits to be produced for each benefiting party, a "conditions" statement that lists the constraints that one must abide while achieving the goal, and a "success criteria" statement that lists the benchmarks that define success (Exhibit 1). Exhibit 2 presents an example of a goal.

Exhibit 1. Components of a Task Goal

To:	**The summary result to be produced**
For:	**Who is to benefit**
By:	**The task that produces the result**
So That:	**The benefits to be produced**
Conditions:	**The circumstances that must exist prior to beginning performance**
Success Criteria:	**The benchmarks that define success**

[1] The contents of this paper apply equally to any assignment made to oneself or by another to you whether it is in the form of a task, a process (a set of tasks that work together to produce a result), or any higher order integration of purposeful activity.

[2] For machine implemented tasks, substitute "objects, processes, or other applications" for individuals or groups.

Building a Useful Goal Statement

Exhibit 2. An Example of a Complete Goal Statement

Component	Contents	Example
To	States the result the performer[1] must produce for the primary beneficiary[2] of the goal. Always begins, "To" and ends with naming whatever product or service outcome accomplishing the goal must generate.	To: "To reduce the accidental programming mistakes produced when using the Netlink API"
For	States for whom the result is being produced—the beneficiaries of the goal.	For: Kernel programmers, customers, owners, and all other stakeholders
By	Names the task that the performer will implement to produce the expected result.	By: Using the Linux Generic Netlink API
So That	States the benefits the result should produce for each beneficiary named in the "For" component. Identifies how the beneficiaries of the goal will be better off once the "To" is achieved.	So That: ■ Programmers focus more on the problem to be solved and less on correcting problems caused by inherent complexities of using the Netlink API such as Namespace collision of protocol families and Demultiplexing messages. ■ Customers get a better product, faster. ■ Owners have lower development costs and shorter development cycle times
Conditions	States other requirements the performance of the task must satisfy. These may: ■ limit resources used (e.g., time, money, people), ■ require a performer to do certain actions—e.g., use a certain tool or involve specified people or ■ set as "off-limits" certain decisions or actions.	Conditions: ■ Must use the 2.6 series kernel. ■ Must register a user process with the kernel component for it to send unrequested messages or data to the user process ■ Must recognize that, in user space, the API is socket based and program accordingly
Success Criteria	The benchmarks that must be met for the goal to be judged as achieved. Always includes criteria that test whether performance of the task: ■ produces the result specified in the "To," ■ delivers the benefits specified in the "So That," and ■ satisfies the constraints recorded in the "Conditions."	Success Criteria[3] ■ Programming mistakes related to using the Netlink API are reduced ■ Programmers' time spent focusing on the problem to be solved relative to fixing Netlink API coding errors increases ■ Quality of code produced using Netlink API is elevated ■ Development time for code using Netlink API is produced faster ■ Development costs for components using Netlink API are reduced ■ Programmers apply the guidance only with code using the 2.6 series kernel ■ Code registers with the kernel component every user process that must be sent unrequested messages or data ■ Coding is consistent with the understanding that the API is socket based

[1] The performer is the person responsible for accomplishing the goal.
[2] The primary beneficiary is the person or group listed in the "For" component of the goal whose absence from the list would render the goal unimportant to pursue.
[3] The statements recorded here are abridged versions of the success criteria. Each has the anchor of the success criterion and the meaning of its target. They are used to accommodate the space available in a goal statement. See Exhibit 8, page 7, for the complete statement of success criteria. Each goal statement should have a such a table attached to it.

Building a Useful Goal Statement

Representing a Goal

Exhibit 3. Representing a Goal

1. Draft the goal statement
2. Test its correctness.
3. Document the finalized goal.

To represent a goal, draft each component of the goal statement (Exhibit 3). Use the clarifying questions presented in Exhibit 4 to assist you. Test the draft logically and with all persons knowledgeable of the task whose purpose the goal represents. Use the guidance provided in Exhibit 5, next page, to assist you. Once done, make any corrections necessary and document the goal in its final form.

Exhibit 4. Suggested Clarifying Questions for Drafting a Goal Statement

Component	Clarifying Question
To	When this assignment is done, what ultimate, tangible result (output or outcome) should be produced?
For	Who is to benefit from the result this task produces?
By	How shall I name the task that accomplishes this goal?
So That	What benefits should this task produce for each benefiting party?
Conditions	■ Is there a cycle time within which this task must be completed? ■ Where or when may this task be done (settings, contexts)? ■ Where or when may this task *not be* done (settings, contexts)? ■ With whom or what should the task performer coordinate during task implementation? What information should be exchanged? What communication protocol should be followed? ■ What access to information is permitted during task performance? ■ What access to information is *denied* during task performance? ■ What resources will the performer have to work with in getting the task done? *Tip:* Consider people, training, tools, equipment, facilities, budget, etc.[1] ■ What resources are off-limits to the performer? ■ What authority will the performer have for making decisions about direction, approach, resourcing, and the application of resources during task execution? ■ What authorities are *denied* to the performer during task execution?
Success Criteria	How should the successful achievement of this goal be judged? ■ What features of each results should be measured? How? What values must be found to claim success? ■ What features of each promised benefit should be measured? How? What values must be found to claim success? ■ What features of each condition should be measured? How? What values must be found to claim success? *(For more assistance in specifying success criteria, below)*

[1] Adjust concepts for machine performance—e.g., accessible RAM, available storage space, type of input or output media.

Building a Useful Goal Statement

Exhibit 5. Verifying the Correctness of a Complete Goal Statement

Component	Contents	Test
To	States the result the performer[1] must produce for the beneficiary of the goal. Always begins, "To" and ends with naming whatever product or service outcome the task must generate.	If the "To" represents a result that the primary beneficiary[2] confirms as what he, she, or they seek, then the "To" is verified.
For	States for whom the result is being produced—the beneficiaries of the goal.	If the "For" identifies the primary beneficiary and every other party who is to benefit from the goal's achievement, then it is verified.
By	Names the task that the performer will implement to produce the expected result.	If the "By" specifies an action that one might reasonably expect will produce the result specified in the "To" given the "Conditions" specified in the goal, then it is verified.
So That	States the benefits the result should produce. Identifies how the beneficiaries of the goal will be better off once the "To" is achieved.	If the "So That" lists a benefit that represents accurately the expectation each named beneficiary seeks to realize from the achievement of this goal *and* each benefit is a likely outcome of accomplishing the result listed in the "To," then it is verified. (Note: If the first condition is passed and the second is failed, then fix the "To." If both are failed, then fix the "So That" and retest.)
Conditions	States other requirements the performance of the task must satisfy. These may: ■ limit resources used (e.g., time, money, people), ■ require a performer to do certain actions—e.g., use a certain tool or involve specified people, or ■ set as "off-limits" certain decisions or actions.	If the "Conditions" state all the constraints that apply to how the goal must be achieved and specify all the resources needed for achieving this goal, then it is verified; otherwise, correct the conditions and retest the "By."
Success Criteria	The benchmarks that must be met for the goal to be judged as achieved. Always includes criteria that test whether performance of the task: ■ produces the result specified in the "To," ■ delivers the benefits specified in the "So That," and ■ satisfies the constraints recorded in the "Conditions."	If the "Success Criteria" list an anchor that reflects the achievement of the "To," the production of each benefit listed in the "So That," and the satisfaction of each conditions *and* each anchor has a target associated with it *and* each target for success is sufficient to satisfy the beneficiaries of the goal and each constraint applied to accomplishing it, then it is verified; else, correct the success criteria as needed and retest.

[1] The performer is the person responsible for accomplishing the goal.
[2] The primary beneficiary is the person or group listed in the "For" component of the goal whose absence from the list would render the goal unimportant to pursue.

Defining Success Criteria

Success criteria tend to be the most difficult goal component to define. As stated above, they are the benchmarks that task performance must meet for it to be judged successful. You need a success criterion for judging whether you have:

- produced each expected output,
- provided each benefit your task was to deliver, and
- satisfied every condition with which your performance was to comply.

Each success criterion is composed of three elements—an anchor, a measure, and a target (Exhibit 6). Exhibit 7 provides one examples of a success criterion. The remainder of this guidance will describe further each component and build the success criteria for the goal specified in Exhibit 2.

Exhibit 6. Components of a Success Criterion

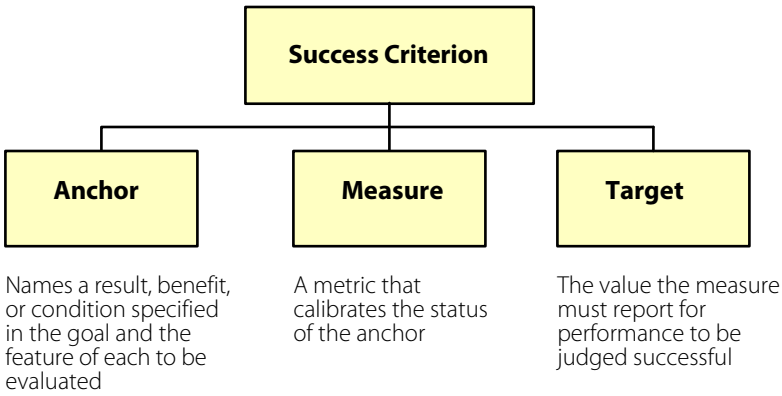

Exhibit 7. An Example of a Success Criterion

Anchor	Measure	Target
Tells what the evaluator wants to know the status of	*Tells how the evaluator gauges status*	*Tells the value on the measure that defines success*
Revenue growth (net of inflation and price increases)	Subtract current quarter revenues from revenues earned in the same quarter last year. Divide by revenues earned in the same quarter last year and multiple by 100%.	12%
Together, the components provide an evaluator the information needed to judge whether the element of a goal specified in the anchor has been successfully accomplished.		

Building a Useful Goal Statement

Anchors

The anchor identifies a focus for evaluation—the who or what the evaluator will measure the status of. It names a result, benefit, or condition specified in the goal and the feature of it to be evaluated. For example, given the goal specified in Exhibit 2, the possible anchors would be:

- Programming mistakes related to using the Netlink API,
- Programmers' time spent focusing on the problem to be solved relative to fixing Netlink API coding errors,
- Quality of code produced using Netlink API,
- Development time for code using Netlink API,
- Development costs for components using Netlink API,
- Programmers apply the guidance only their code uses the 2.6 series kernel,
- Registration of user processes that must be sent unrequested messages or data with the kernel component, and
- Consistency of coding with the understanding that the API is socket based.

Typically, more anchors are specified in a goal than the goal's manager will evaluate. How many anchors have success criterion written for them depends on the importance of placed on correct goal achievement by the business and the cost associated with completing the evaluation. An anchor's importance presumes that someone will make meaningful decisions based on its evaluation. Clearly, if no one will make meaningful decisions based on the evaluation of a particular anchor, then its evaluation is waste.[3]

Measures

The measure tells how you will detect the status of the anchor. Each measure specifies a method for calibration and the time period for making the calibration—e.g., count the number of work processes for which standards are documented monthly, sum the dollar amount of cost savings produced annually, compute the average level of satisfaction with training people report quarterly, or determine the percentage of actions finished on or before their “complete by” dates annually.

A measure may calibrate status in terms of quantity, quality, timeliness, or efficiency.

- Use quantity metrics to register whether some expected result exists, how much or how many units are produced, or how completely some set of activities are done or components of a product are implemented.
- Use quality metrics to register how correctly or how usefully some product or service outcome is rendered; how consistent it is with policy, guidelines or specifications; how satisfying it is to its recipients; or how improved it is over time.

[3] Naturally, if someone *should be* making decisions based on the anchor's status, as judge by his or her fiduciary responsibilities, then that person's behavior should be adjusted and the anchor included.

Building a Useful Goal Statement

- Use timeliness metrics to register whether schedules are met or resources are available when needed.
- Use efficiency metrics to register whether an expected rate or ratio is realized—e.g., the number produced per unit of time or the relationship of monetary cost and benefits either to each other (e.g., cost benefit ratio) or relative to some benchmark (cost effectiveness ratio).

Exhibit 8 suggests which metrics are useful to consider given the focus of your anchor. Exhibit 9, next page, lists the measures used to calibrate the status of anchors specified for the goal depicted in Exhibit 2.

Exhibit 8. Examples of Metrics for Anchors Having Different Focuses

If the Anchor Focuses on...	Then Consider These Metrics...	
The presence or absence of an object or feature	▪ "Yes" or "Present" and "No" or "Absent"	
The amount or size of something	▪ Count ▪ Area ▪ Weight	▪ Length, width, or height ▪ Volume ▪ Sum
The typical amount of something produced multiple times	▪ Mean (or average) ▪ Median	▪ Mode
The degree to which a quality indicator is realized or improvement has occurred	▪ Percent ▪ Difference from expected	▪ Difference from baseline ▪ Slope of the line of best fit for a time series
The timeliness of occurrence	▪ Percentage of on-time deliveries	▪ Difference between expected and actual delivery dates
The efficiency of performance	▪ Cost per unit produced ▪ Number produced per unit of time ▪ Ratio of cost to benefits (ROI)	▪ Ratio of old to new cost ▪ Ratio of unit cost to a benchmark (e.g., industry average)

Targets

The last element of a success criterion is the target. It states the value on the measure that indicates whether the anchor's status meets expectation as defined by the goal—e.g., 25 work process standards documented, $5,000,000 in cost savings realized, "expenditures do not exceed budget" or "report contains all requested information." Exhibit 9, next page, presents an example of a complete success criteria for the standards section of the goal depicted in Exhibit 2.

Exhibit 9. Success Criteria for the Standards Section of the Goal Depicted in Exhibit 2

Anchor	Measure	Target
Programming mistakes related to using the Netlink API	Ratio of the average number of coding errors related to Netlink API use observed when coding does not use the guidance as compared to when it is used	Ratio is greater than 1
Programmers' time spent focusing on the problem to be solved relative to fixing Netlink API coding errors	Ratio of average percentage of development time for code using Netlink API spent fixing Netlink API coding errors to baseline established prior to use of guidance	Ratio is less than 1
Quality of code produced using Netlink API	Ratio of the average incidence of coding errors for components using Netlink API before introduction of this guidance to the average for components using Netlink API after the introduction of this guidance[1]	Ratio is greater than 1
Development time for code using Netlink API	Ratio of the average development time for components using Netlink API before introduction of this guidance to the average for components using Netlink API after the introduction of this guidance[1]	Ratio is greater than 1
Development costs for components using Netlink API	Ratio of the average development cost for components using Netlink API before introduction of this guidance to the average for components using Netlink API after the introduction of this guidance[1]	Ratio is greater than 1
Programmers apply the guidance with code using the 2.6 series kernel	Count the incidences where the guidance is applied with code not using the 2.6 series kernel	0
Code registers with the kernel component every user process that must be sent unrequested messages or data	Count the incidences where a user process that must be sent unrequested messages or data is not registered with the kernel	0
Coding is consistent with the understanding that the API is socket based	Count the incidences of coding statements that are inconsistent with the understanding that the API is socket based	0

[1] Comparison adjusts for the average size and complexity of the components being compared.