A Goals-Means Task Analysis method [1]

Erik Hollnagel

(This is a text recovered from a report to ESA-ESTEC in a project on Task Analysis Methods – 1991.)

1. The Logic Of Task Analysis

The purpose of a task analysis is to break the task into its constituent parts in order to describe the parts as well as the ways in which they are organized in the task. The basic organisation principles must be that each step of the task brings the acting agent closer to the stated goal. This will therefore also be the principle of decomposition. Each task step is assumed to be carried out by an agent. The agent may be an operator (local or remote) or an artefact, e.g. a robot or a computerized function. If a task step does not bring the agent closer to the goal, it is clearly inappropriate and should be avoided or not be prescribed. It follows that a task must be described with reference to the goal of the task. We may here apply the following definitions:

- Goal : A goal is a specified condition or state of the system which signifies the accomplishment of a task as a set of task steps. A goal is what the agent intends to do or to achieve. A description of the goal must include the criteria of achievement, i.e., the conditions that define when the goal has been reached.
- Task : A task is an organised collection of actions carried out by an agent in order to achieve a goal or an objective. The actions are referred to as task steps. The organisation of the actions may be derived from the fact that the individual actions together serve to accomplish the goal or may be prescribed by a procedure - either as a pre-defined sequence of steps, a pre-defined organization principle, or specified dependencies between steps.

A task analysis denotes a method by which a description can be developed of the task steps that will lead to a stated goal as well as the principles by which they are organized. Each task step can be seen as part of the means to achieve the goal. The basic method in the GMTA is a recursive decomposition based on a goals-means relationship. The GMTA is based on cognitive systems engineering (Hollnagel & Woods, 1983) and cognitive task analysis (Woods & Hollnagel, 1987). It further has a strong resemblance to the underlying principles of multilevel flow modelling (MFM, cf. Lind, 1990).

1.1 Goals And Means

In task analysis one can distinguish between two different situations. In the simplest case the task steps can be carried out without imposing further conditions on the system. The steps of the task can be described as a simple sequence, either in a strict linear sequence or with branch points (decision points), i.e., as a tree. If the task steps are single or elementary actions, the result is a non-hierarchical task description. If the task steps are composed of other task steps (aggregation) then the result will be a hierarchical task description; it simply

1 The figures to the original report were made in GEM, and it has not been possible to recover them.

means that the task description includes elements at different levels of elaboration, from global descriptions to detailed elementary task steps. Most task descriptions will be hierarchical, for the very simple reason that we are forced to do so by the limitations we encounter (size of paper, span of memory), as well as the way in which we intuitively organize work.

The task thus describes or points to the means by which the goal can be achieved. This can be written as follows:

Goal "The named goal or state which is to be achieved."
Task "The named task which is sufficient to achieve the goal."

An elementary step is defined as a step that is not decomposed at the current level of analysis. The category is thus not absolute. In practice any given application will have some steps that commonly are considered truly elementary, in the sense that one can safely assume that they can be performed no matter what the conditions are. These are usually the basic skills that every person has qua being a person or qua being qualified for the work in question.

1.2 Preconditions

In the other, more complicated, case the task steps cannot be carried out unless certain conditions are met. It is not enough that a task step will bring the agent closer to the goal; one must also consider the preconditions of the task steps. The task analysis must therefore clarify under which conditions the task steps can be carried out and under which conditions they cannot. These conditions can either be the preconditions that must exist when the task step is started and/or the conditions that must prevail during the execution.

Examples:

In order to start a car there must be voltage on the battery; but the battery need not even be present when the car is running. The presence of a battery (or an alternative source of energy) is therefore a precondition for starting the car. This could be written as:

Goal The car has been started.
Task Start the engine by turning the ignition key.
Precondition Battery is present.

(The indentation is used to show a change in the level of description, i.e., that this task is subsumed under the previous precondition.)

In order to repair the roof of a house, a ladder must be present (to get on to the roof). The ladder is, however, not required during the repair (although it may be useful for getting down again, hence completing the task). This can be written as:

Goal The roof has been repaired.
...
Goal Person is on the roof.

Task Climb the ladder

Precondition Ladder is present.

Conditions that must be present when a task is started obviously need not be present during execution. A related question is whether conditions that must exist during execution also must be present when a task is started. Although it may seem so in many cases, it is not difficult to find examples of the contrary. If, for instance, I want to drive in my car from Copenhagen to Paris, I need enough petrol to do so. But I do not need to have it all when I start the journey, since I can refuel on the way. Refuelling thereby becomes a sub-goal. An aircraft, on the other hand, usually need to have all its fuel on board when it takes off. If the task is to photograph a solar eclipse, then the eclipse should clearly not be there when the task or mission is started. Despite these counter-examples it is probably more often than not the case that execution conditions must be there from the start.

1.3 Establishing Preconditions

If the preconditions are not true (if they do not obtain) when the task step is to be carried out, they must be established; this, then, defines a new task, the purpose of which is to establish the preconditions and thereby enable the (preceding) task step. The new subsidiary task will not necessarily directly bring the agent closer to the goal, but rather serves as an enabling task. However, in so far as the goal of the enabling task is to establish the conditions that makes it possible to carry out the preceding task step, the enabling task does, in some sense, bring the agent closer to the goal. The notion of a precondition can be expressed as follows:

Goal	"The named goal which is to be achieved."
Task	"The named task which is sufficient to achieve the goal."
Precondition	"The condition that must be fulfilled before the task can be carried out."
Task	"The named task which is sufficient to establish the preconditions."

The Goals-Means decomposition is a principle that requires the task analysis to be recursive. If the task steps can be described without preconditions as a linear sequence (including as a tree), the task analysis need only be repeated rather than recursive, i.e., there is nesting of actions but no nesting of goals and sub-goals. The recursive analysis will, of course, automatically produce a hierarchical result.

1.4 Execution Conditions

In addition to the preconditions there are also some conditions which must be maintained during the execution of the task step. An execution condition can either be the same as a precondition, or be a separate condition. The execution conditions will typically be of two types: either the availability of the resources necessary to carry out the task steps, or conditions that refer to other task steps. An execution condition could, for instance, be that two tasks were completed at the same time or - more stringently - that they were carried out during the same time interval or even simultaneously. In the majority of cases execution conditions are not stated explicitly, but are rather assumed by or inferred from the description of the task step. Thus if the task step is to paint the roof, an execution condition is that there is

a constant supply of paint. But since it is very difficult to paint something without having paint, the execution condition is not strictly necessary.

1.5 Postconditions

Formally speaking, one can consider each task step as producing a postcondition (a post execution condition) or an effect. (There are actually two types of postconditions; the distinction between them will be explained below.) A precondition will always refer to a set of task steps which have to be carried out if the precondition is not met. The outcome of the task steps (the postcondition) is what establishes the precondition. Task steps should be defined so that this postcondition will either be achieved (TRUE) or not achieved (FALSE), according to some criterion. The task steps themselves can therefore also be described in terms of these conditions, i.e., that a task step is only considered completed if it has the appropriate effect (or returns the condition TRUE). Otherwise the analysis must be taken a step further (by decomposition) in order to achieve the desired postcondition. Clearly, if the postcondition for a task step is not achieved (it remains FALSE) and if there are no ways in which it can be changed, then the goal cannot be achieved and the task step therefore has failed.

This, by the way, provides us with another definition of an elementary task step, as being one which always produces the required effect (returns the condition TRUE). In other words, it is a task step which for the sake of the analysis can be assumed always to be executable.

In addition to the effect, a task step can also produce another type of postcondition which are called side-effects. A side-effect is a change which is not among the intended effects of the task step. Side-effects may be either positive, neutral or negative. A positive side-effect produces consequences which improve the overall execution of the task, e.g. by rendering later preconditions superfluous. A neutral side-effect does not affect the execution of the task. A negative side-effect produces consequences which are detrimental to the execution of the task, e.g. by cancelling the effect of earlier task steps, by creating needs for new preconditions or blocking planned task steps, etc. A task analysis should clearly try to anticipate negative side-effects and avoid them if possible.

These principles constitute the basic structure of task analysis, and all variations can be derived from that - either as a question of techniques for analysis or representation, or as a question of focus.

2. Task Analysis versus Performance Analysis

The term task analysis is often used as a generic label. Any survey of what commonly is called task analysis methods will quickly show that they represent many different meanings of the term. A little closer inspection will, however, reveal that they fall into a few main categories. Some concern the analysis and description of working situations which are not yet in existence; some concern the description and analysis of observations of work being carried out; and some concern various ways of further analysis or presentation of data about tasks (from either source).

We shall use the term task analysis to refer to the analysis and description of tasks based on information about the system (specifications and detailed design data) rather than based upon observations. Task analysis in this sense is carried out as a part of the system design, and can serve several purposes. In our case the primary purpose is to provide input data for the HRAM; other purposes could be preparation of procedures, detailed design of support

systems or man-machine interface, etc. Task analysis describes what the tasks should be (or are expected to be) and is in this sense prescriptive, although usually not in the meaning of being normative. (A possible execption are procedures which normally are supposed to be followed to the letter.) It is a prescription of the general principles rather than of the minute details.

Task analysis is seen in contrast to methods of task description or performance analysis (the latter term is preferred). A task description produces a generalized representation of activities as they have been carried out. It is based on empirical data or observations rather than on design data and specifications. A typical example is link analysis or even HTA. Properly speaking, task description or performance analysis deals with actions rather than with tasks. (According to the French tradition in ergonomics, a distinction is made between tasks and actions (activities). Actions are the instantiations of the tasks, i.e., the actual performance, whereas the tasks are the description of the intended/required actions. Another possible term would therefore be action analysis.)

The final set of methods have to do with specialized analysis and/or (re)presentation of the outcome; they shall accordingly be called task presentation methods. Examples of that are workload analysis, timeline analysis, operational sequence diagrams, critical path analysis, etc. The common characteristics of these methods are firstly that they are not really task analysis methods in the meaning defined here and, secondly, that they can be applied to data about tasks whether they are derived from design specifications or empirical observations. Thus the Goals-Means Task Analysis described in the following does not in itself point to any specific way of presenting the outcome, and need therefore be supplemented by a suitable way of task representation.

The relation between the three categories of task analysis methods is shown in Figure 1.

#

Figure 1: Task Analysis And Performance Analysis.

3. A Goals-Means Task Analysis Method

3.1 The Basic Concepts

The Goals-Means Task Analysis Method, hereafter referred to as the GMTA, uses the following basic concepts: goal (Goal), task or just task step (TaskStep), precondition (PreCon), execution condition (ExCon), and postcondition (PostCon). We will first provide the basic definitions of these concepts, and later, by means of examples, show how they can be applied. The example used to illustrate the definitions is the task called “EVA Inspection” described in ???

3.1.1 Goal

A goal describes the state that is to be achieved by the acting agent (usually a human operator but possibly also an intelligent or autonomous artefact). In the example introduced below “Astronaut is outside spacecraft” constitutes a goal. To achieve a goal it is necessary to carry out a TaskStep.

Goals are either defined as derived goals or independent goals. The difference is whether the goal is the result of the decomposition of another (parent) goal or not. The former are called

derived goals (or sub-goals) and the latter are called independent goals or top-goals. A pre-condition is equivalent to a sub-goal. A sub-goal always refers to a parent goal, and when the sub-goal has been achieved control returns to the parent goal. When the top goal has been achieved the task is for all practical purposes completed.

The distinction between derived and independent goals is not absolute but pragmatic, and depends on the chosen level of description. Just as some task steps are considered elementary in a practical sense (cf. the discussions above), so some goals will be considered as independent goals or top goals. Unless this is done there will be infinitely many levels of recursion.

3.1.2 Task Step

A goal is achieved by a set of task steps. This set need have only a single member. The term task will be used loosely in the following; but the terms step, and set of steps, will be used to refer precisely to the steps or sequence of actions that are attached to a goal.

A task step can be composed of other task steps in which case it serves as a label or reference to a set of task steps (a routine, a group, a procedure). This is usually done merely for notational convenience, as discussed above in relation to hierarchical task analysis. Each set of task steps may be concatenated as a larger task step; conversely, each larger task step can be decomposed into the constituent task steps.

Task steps describe activities which can be carried out, i.e., where it is assumed - for the sake of the analysis - that the pre-conditions are true. This assumption can, of course, always be challenged and the analysis taken one step further. In this way a task step which has a pre-condition will create a sub-goal which expresses the pre-condition.

3.1.3 Precondition

A precondition (PreCon) describes the conditions that must be satisfied for a task step to be carried out. When a precondition is found, it is considered as a sub-goal (a derived goal). It is this feature which introduces the recursiveness of the GMTA. A precondition corresponds to a sub-goal and can be followed by one or more task steps which, when carried out, will achieve the precondition. In generic terms a representation would be:

Goal “The named Goal that is to be achieved.”
Step1 “The name of the (first) task step that can achieve the goal.”
PreCon1 “The condition that must be fulfilled before the task can be carried out.”
Step1.1 “The first step towards fulfilling the first condition.”
Step1.2 “The second step towards fulfilling the first condition.”
PreCon2 “The precondition for the above step.”
Step2.1 “The first step to achieve this new condition.”
Step2.2 “The second (and final) step to achieve this new condition.”
Step1.3 “The third (and final) step towards fulfilling the first condition.”
Step2 “The second step towards achieving the goal.”
...

In order to make the terminology unambiguous only task steps can have preconditions.

3.1.4 Execution Conditions

An execution condition describes the conditions that must be satisfied while a task step is carried out. An execution conditions is thus always attached to a task step. In cases where the execution condition is implied by the description of the task step, it need not be defined explicitly. Most execution conditions will therefore refer to temporal conditions and relations to other task steps.

3.1.5 Postcondition

A postcondition (PostCon) describes the conditions which are not defined or included by the goal or task step, but which may occur anyway. Postconditions refer to the side-effects which are deemed to be important because they are known or assumed to have consequences for e.g. other preconditions - either in a positive (synergistic) or negative (antagonistic) way. In this context the postconditions are therefore different from the (main) effects. Practically all actions will have some side-effects, although most of these safely can be neglected. (That is the case, at least, for normal conditions. The art of risk and reliability analyses is, of course, to to know when a side effect should be neglected and when it should not.)

3.1.6 Timing Conditions

An additional and important concept is the timing or sequence conditions of a task step. Sequence conditions describe constraints on the sequencing (starting and stopping) of a task step. The sequence conditions can be expressed as relative timing conditions using e.g. the suitable terminology proposed by e.g. Allen (1983). According to this any event can be described in terms of four self explaining time indicators:

- Earliest Starting Time (EST),
- Latest Starting Time (LST),
- Earliest Finishing Time (EFT), and
- Latest Finishing Time (LFT).

It is clear that these timing conditions refer to an ordinal rather than an interval scale. They do neither require estimates of duration nor references to measured time (absolute or elapsed). Such information is rarely available until the last stages of system development, but may of course be used if there. Even ordinal timing conditions are, however, extremely useful to provide further details in the description of pre-conditions or execution conditions. A timing condition could, for instance, be that the beginning of a task step must take place after the ending of another task step, or that two task steps must be carried out simultaneously:

```
...
Step: Do X
Step: Do Y
PreCon: ESTstepX GT LFTstepY
...
```

Indications of simultaneous execution of task steps is relevant mostly for steps belonging to different tasks (i.e., with different but possibly related goals). Time indications may be given as absolute time (clock time) or relative to other events, if this information is available to the analyst. An example is:

...
Step: Do X
Step: Do Y
PreCon: LSTthis step - max(EFT, LFT)previous step < 2 minutes
...

In this example the pre-condition directly describes how the condition is going to be determined and there is consequently no need for other task steps. Although a sequencing/timing pre-condition functionally is equivalent to a sub-goal, being a condition which must be fulfilled, it does not make sense actually to transform it into a sub-goal since there usually are no explicit task steps which will accomplish it.

Timing conditions are used for either predoncitions or execution conditions. A timing condition could, for instance, be that the beginning of a task step must be after the ending of another task step, or that two task steps must be carried out simultaneously. Time indications may be given as absolute time (clock time) or relative to other events.

3.2 An Example

The minimal description of a task must contain a goal and a task step, where the task step is an elementary step as defined above. An example is:

Goal: Computer is running.
Step: Start the Computer.

In most cases there will, however, be more than one task step needed to achieve the goal, as well as some pre-conditions for the goal. The above example would be:

Goal: Computer is running.
Step: Start the Computer.
PreCon: Power Source Is Connected.

where the task step actually refers to a separate task, defined as follows:

Goal: Computer is running.
Step: Turn the Computer On.
PreCon: Power Source Is Connected.
PreCon: Make Sure There Is No Diskette In Drive A.
Step: Press Enter In Response To the Date and Time Prompts.
Step: Confirm That the MS-DOS Prompt Is Correct.

Depending on what one wants to assume as an elementary task step, this description may either be accepted as adequate or developed further. The first pre-condition may be extended with a task step to be carried out if the pre-condition is false:

Goal: Computer is running.
Step: Turn the Computer On.
PreCon: Power Source Is Connected.
Step: Connect power source.
PreCon: Make Sure There Is No Diskette In Drive A.
Step: Press Enter In Response To the Date and Time Prompts.
Step: Confirm That the MS-DOS Prompt Is Correct.

The second pre-condition may be replaced by a task step which expresses a condition and an action:

Goal: Computer is running.
Step: Turn the Computer On.
PreCon: Power Source Is Connected.
Step: Connect power source.
Step: If there is a Diskette in Drive A, then remove the Diskette from Drive A:.
Step: Press Enter In Response To the Date and Time Prompts.
Step: Confirm That the MS-DOS Prompt Is Correct.

This procedure can, of course, be repeated, depending on time and need.
>>An additional example is found in the documentation from the test case<<

4. Formalization Of The GMTA Method

The relation between the basic building blocks of a GMTA is shown in Figure 2. (Note that this is not intended to depict the control structure of a task.) This will serve as a useful anchoring point in describing the method in a step-by-step fashion.
#
Figure 2: The Basic Building Blocks Of Goals-Means Task Analysis.
The minimal description of a task must contain a goal and a task step, where the task step is an elementary step as defined above.

4.1 Detailed task steps In The Goals-Means Task Analysis Method

The steps in the GMTA can be described as follows.

- For the given task domain (application domain), identify the top goal, or top goals if there are more than one.
  - For each top goal and for each goal, do the following:

        - Give the name of the goal. (It is useful to start the name with a noun.)
        - Describe the goal.
        - Identify and name the attached set of task steps. For each task step mark if it is decomposable or elementary.
    - For each task step, do the following:
        - Give the name of the task step. (It is useful to start the name with a verb.)
        - Give the name(s) of the relevant goal(s) or pre-conditions, i.e., the goals or pre-conditions that points to this task.
        - Define and describe the constituent set of task steps or (sub)goals. For each task step mark if it is decomposable or elementary. For each goal produce a description of it.
    - For each pre-condition, do the following:
        - Give the name of the pre-condition. (It is useful to start the name with a noun.)
        - Give the name(s) of the relevant task step(s), i.e., the task steps that have this pre-condition.
        - Define and describe the enabling set of task steps. For each task step mark if it is decomposable or elementary.
    - For each execution condition, do the following:
        - Give the name of the execution condition. (It is useful to start the name with a noun.)
        - Give the name(s) of the relevant pre-conditions or task step(s), i.e., the pre-conditions or task steps that have this execution condition.
        - Define and describe the enabling set of task steps, if any. For each task step mark if it is decomposable or elementary. Otherwise describe the sequence (timing) conditions.
    - For each post-condition / side-effec), do the following:
        - Give the name of the post-condition / side-effect. (It is useful to start the name with a noun
        - Give the name(s) of the relevant task step(s), i.e., the task step(s) that refer to this post-condition / side-effect.
        - Describe the details of the post-conditions / side-effects.

5. Notes On The Implementation Of The Method

The completeness of applying the GMTA can be checked in the following way:

- For each goal except the top goal(s) there must be a parent goal. For each goal there must be a goal description. Goals can either be defined independently or from task steps and from pre-conditions.
- For each task step there must be a task step description. Task steps are defined from goals or from pre-conditions.

- For each pre-condition there must be a pre-condition description. Pre-conditions are defined from task steps.
- For each post-condition or side-effect there must be a post-condition description. Post-conditions are defined from task steps.

6. References

Allen, J. (1983). Maintaining knowledge about temporal intervals. Communications of the ACM, 26, 832-843.

Hollnagel, E. & Woods, D. D. (1983). Cognitive systems engineering: New wine in new bottles. International Journal of Man-Machine Studies, 18, 583-600.

Lind, M. (1990). Representing goals and functions of complex systems. An introduction to multilevel flow modelling. Technical University of Denmark, Institute of Automatic Control Systems.

Woods, D. D. & Hollnagel, E. (1987). Mapping cognitive demands in complex problem-solving worlds. International Journal of Man-Machine Studies, 26, 257-275.