

# TaskArchitect

## An Introduction to Task Analysis using TaskArchitect (TaskArchitect を使った課題分析入門)

*WHITE PAPER*  
(白書)

著者：Jon Stuart (ジョン・スチュアート)

### 【おことわり】

図表は「TaskArchitect (R) Task Analysis Software - Taking the work out of Task Analysis - User Guide / Task Architect Inc. / Windows Version 1.023, Manual Version 1.1」から転載しました。

This document was translated by Japan Organization for Employment of the Elderly and Persons with Disabilities (JEED) as part of their research into the application of task analysis.

## 目次

---

はじめに	P.2
課題分析とは何か	P.3
課題分析の利点	P.4
課題分析の用途	P.4
課題分析を使用する時期	P.6
利用できるのは誰なのか	P.6
課題分析の概要	P.8
ステップ1：何をどう分析するのか	P.10
分析の文脈	P.10
要求される課題詳細度	P.10
記録する必要のある課題属性	P.10
ステップ2：情報の収集	P.11
停止規則の適用	P.12
ステップ3：収集された情報を検証する	P.13
ステップ4：課題分析の表記法の応用	P.14
課題記述	P.14
プラン	P.15
課題属性	P.20
課題の番号処理	P.23
ステップ5：分析結果の表示	P.25
ステップ6：結果の利用	P.30
ステップ7：課題分析の更新管理	P.31
課題分析の種類	P.31
職務分析	P.31
認知的課題分析	P.33
参考図書	P.34

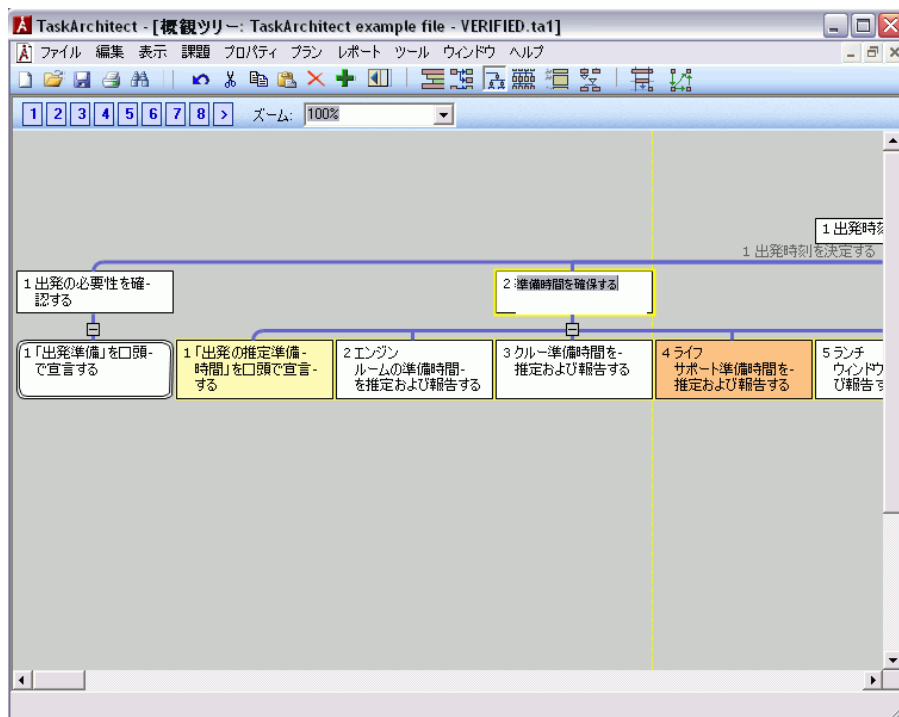
はじめに

*TaskArchitect* は課題分析の支援専用設計されている初の市販ソフトウェアです。課題分析にはおそらくその実務家の数だけのアプローチがあるはずですから、きわめて柔軟なツールになるように設計されています。

本書は、現在課題分析の専門家ではない方々のために書かれており、複雑な課題を分析する際にそれがどのように役立つかについて説明しています。本書はまた、*TaskArchitect* が課題分析をどのように支援するかの例を提示し、わずかのキーストロークで、これまで困難だった作業にいかに対処できるかを示しているのです。より経験豊富な実務家にとっても情報源になりえるはずです。

課題について構造化された膨大な量の情報を取り込む能力、ユーザとその環境、課題のリストと課題の相互関係ダイアグラムとの間の迅速な移動、課題の自動番号変更と変更後のプラン、種類の豊富なレポートと出力のオプションなどを含め、*TaskArchitect* は専用課題分析ツールが備える全ての利点を提供します。課題分析はプロジェクトにとって大きな価値を秘めているので、ツールの構築は行われてきました。しかしこれまでのところ、結果を記録し、表示し、分析するために要する労力のため、大規模な軍事プロジェクトや安全性が最重要視されるプロジェクトを除けば実務上の障害になってきました。

*TaskArchitect* なら設計プロジェクトにおいて設計チームの誰にとっても課題分析が現実的な作業になります。ビジネス上の問題を分析し解決するために課題分析がどのように使用されていたかの例については *TaskArchitect* の Web サイトに掲載されています。



## 課題分析とは何か

---

課題分析とは、人々が課題をどのように実行するかについて情報を収集し、文書化する際の構造的アプローチです。課題には職務の遂行やツールの使用又は最終目標の達成に向けられた活動などがあります。分析の目的は、ユーザとその訓練、ツール及び職場の適合性を良くするために活動の実行に正確に何が係わるのかについての理解を深めることにあります（「成果」）。分析では最低でも、目標を達成するために実施されるステップ（課題と呼ばれる）とそれらがどれだけうまく調和するかを記述するプランを明らかにします。課題はさらに細かい課題に分解することができますが、これらはサブ課題と呼ばれます。活動全体を構成する異なるステップ間の関係を示すため課題、サブ課題及びプランが階層的に配列されます。本書では、「目標」と「課題」という用語が交換可能な同義語として使用されています。実務上、この区別は重要ではありません。しかしながら、理論的なレベルでは、目標は課題の実行によって達成されることからその違いは重要です。

さらなる分析には、任務／職務の条件、基準、履行ステップ、要求される技能と知識、安全性と環境の要因、参照事項、設備、業績測定など、課題の属性についての情報収集が係わってきます。課題分析は設計への組織的アプローチを実施する際の最初のステップになります。課題分析では、現在職場で実際に何が起きているのかについて焦点を合わせることもできれば、新しいシステムの設計を指定するためのツールとしても使えます。課題分析では目標をどう達成するかについての詳細な記述が得られるので、設計担当者は人間の遂行能力を改善するための手段を提供できるようになります。

*TaskArchitect* は、課題を迅速に分類して再配列できる環境及び課題の内容をさらに定義する様々な属性を分析者が定義できる場を提供します。

分析者が過去の経験からその有効性を確認している分析の目的と方法に応じて、課題分析には多くの異なるアプローチがあります。*TaskArchitect* は、分析者が最小限の手間で自分の好きなアプローチを利用できるようにするため柔軟性を念頭に設計されています。

*TaskArchitect* は、主要目標からサブ課題、次にそれぞれのサブ課題からさらに詳細にという具合に課題を階層的に分解する課題分析の手法をサポートします。アプローチにおける初期の段階では課題フローや概念的分析が役立つという人もいるでしょうが、*TaskArchitect* はそれらを生成するには設計されていません。代わりに *TaskArchitect* は、複雑な課題のモデリング用ソフトウェアを使用する前の迅速な課題分析を念頭に設計されています。

本書では一般的なアプローチとして、つまり皆さんが自分のニーズに合わせて調整できるものとしての課題分析について説明しています。また、職務分析や認知的課題分析など、一部の特定の種類の課題分析についても説明しています。

## 課題分析の利点

---

課題分析では目標を達成するために実施される課題についての簡潔で曖昧でない記述のための構造が得られるので、ユーザの行動を記述し、製品設計に与えるその影響について検討するのが容易になります。課題分析では人々がどのように課題を実行するかについての基本的な情報を提供してくれるので、様々な分野の方々がシステム設計にそれぞれの専門知識を効果的に応用できます。

分析者はユーザが使用するツールやシステムの詳細に忙殺される代わりに、ユーザの課題に焦点を合わせることで、ユーザが何を行い、それをうまくやり遂げるために何を必要としているかについて理解するための基盤を構築します。課題を明確な形で把握し、相互の関係を調べることで、このアプローチなら単純化したり、安全性を高めたり、又は学習しやすくすることで設計者が改善することのできる課題を明らかにするのに役立ちます。

*TaskArchitect* は、一連の課題の高い視点からの記述から多くの詳細度や支援情報の取り込みまで、どのような詳細が要求される課題分析でもサポートします。これによって分析者は現在の問題の解決に要求される詳細度だけに分析作業を制限することができます。また分析者はある任意の時点で特定の設計の討論で要求される情報だけが提示されるように詳細の各レベルを隠蔽することもできます。

分析の出力内容を明確且つ簡潔に表示することで、*TaskArchitect* なら課題分析者とユーザとの間の共同作業も支援できます。分析出力を討論のための基本資料として使用することもできれば、分析者とユーザと一緒に分析を進めていくこともできます。どちらのアプローチでも課題関連の情報の収集と分析が迅速になり、情報の検証に要する時間も短縮されます。

## 課題分析の用途

---

ここで取り上げる方法は、皆さんが皆さんの組織で自分の手法を活かしていく際に課題分析がどのように利用できるかについてのアイデア源となるものです。課題分析は以下に記載される業務上の利点を実現するための基本となる情報を提供します。

### 製品、プロセス及びシステムの設計

- 製品の利用における困難を予測し、ユーザの利用状況を予測する。
- ユーザビリティ及び／又は機能上の要件についてシステムを評価する。
- 既存製品の使用における困難さを理解する。
- 製品のための訓練マニュアルを開発する。

- 表示する必要がある不可欠な情報を決定する。
- 製品利用の異なる段階において画面上又はマニュアルで要求される情報を決定する。
- 設計上の異なる選択肢を使用する際の潜在的な困難を予想する。
- 開発者と開発プロセスに係わるその他の者の間のコミュニケーション手段を提供する。
- 設計に対するシステムベースのアプローチのための基本情報を提供する。
- 操作手順を設計する。
- ユーザとマシンに作業を割り当てる。

### **効果的な訓練プログラムの設計**

- 学習の容易さ及びシステム間の知識の移転について評価する。
- 妥当性のない訓練、訓練過剰又は訓練不足について把握する。
- カリキュラム要件を決定する。
- 部署横断的に訓練のための取組みを統合する。
- 訓練ニーズを分析する。

### **職務の設計及び評価**

- 最低限要求される教育と経験について決定する。
- 人事選考基準を定義し、選考試験の草案を作成する。
- チームを組織する。
- 作業量を分析する。
- 職務記述書に作業環境を定義する。
- 人材採用面接の質問事項を明らかにする。
- オリエンテーション用の資料を企画する。
- 報酬水準について決定する。

### **購入の決定**

- 現在及び将来の従業員の能力に対する異なる購入の影響について分析する。
- 異なる設計の安全でない利用に伴うリスクについて分析する。
- 保守活動のコストを予測する。

### **安全性の改善**

- エラー率が許容できないほど高くなる可能性のある課題を予測する。
- ミスによる潜在的な影響について文書化する。
- 安全性が最重要な課題を明らかにする。

## 課題分析を使用する時期

---

課題分析はプロジェクトの開始時に使用される情報を提供し、それは後の設計段階における決定のための基礎となります。

新しいシステムを設計する際に、提案される課題の流れとツールの使用について概略を決めるのに課題分析を使用することができ、最終設計について合意を得るため潜在的なユーザと設計者にそれを評価させることができます。見直しを行った課題分析は、設計、文書担当、訓練の各チーム間でコミュニケーションツールとして使用することができます。

既存システムを改善する際には、改善の余地のある領域を明らかにするため現在のシステムがどのように利用されているかについて文書化するのに課題分析を使用することができます。

訓練のニーズを評価する際には、現在実施されている内容、現在の訓練の有益な点及びより多くの訓練が役立つ領域を文書化するのに課題分析を使用することができます。

課題分析では人々が課題をどのように実行するかについての基本的な情報を提供してくれるので、様々な分野の方々がシステム設計にそれぞれの専門知識を効果的に応用できます。

## 利用できるのは誰なのか

---

課題分析は、システム設計者、ソフトウェア設計者、テクニカルライター、訓練制度の設計者、安全性の専門家、ならびに人事のスペシャリストなど、システム設計に携わる様々な人々によって使用されるアプローチです。本来の能力をユーザが使用しているツールの記述に費やすのではなく、彼らの課題に常に焦点を合わせることができます。*TaskArchitect* は課題の番号処理や課題相互の関係を表示するダイアグラムの作成を引き受けることで、分析者が課題に関する情報を構造化するのを支援し、収集されているデータに関する迅速なレポートを作成する手段を提供します。こうした機能により分析者が利用する技法に簡単に課題分析を組み込めるようになります。

**次の領域での経験は分析の精度と有用性を改善します。**

- どのように作業しているかについて従業員から有益且つ正確な情報を収集するためインタビューを行う。
- 収集される情報が後の設計段階でどのように利用されるかについて理解する。
- 簡潔、明瞭な表現で課題を記述する。
- 課題がどのように調和するかについて明確に記載する。

小さく簡単な活動の課題と課題プランを文書化する単純な課題分析の実行が極めて簡単に行えます。より大きな分析を試みるにつれ、「分析麻痺」を回避するため、妥当なレベルの詳細はいつ記録されるべきか、どの課題に一番重点を置くべきか、分析がプロジェクトにどのような利益を与えるかについてより多くの知識が要求されます。

課題分析を実地に学んでいく際によくあることは、課題に関して分析の最後に役立つ以上の情報（課題属性）を記録してしまうことです。重要な要素については作業の終了間際まで明確に理解されないこともあるので、ある程度、これは分析にはつきものです。これは分析活動における正常な学習曲線の一部でもあります。問題の主な特徴を迅速に見極める能力を養うまでには分析過剰になるのも正常なことだからです。

あらゆる技能の習得と同じように、分析者は慣れによって、問題へのアプローチについて自分自身のスタイルを確立し、より高度な問題に取り組むことができるような表記法やショートカットを生み出していけるようになります。

皆さんの会社で既に定義済みの表記法や規則及び報告手順のある課題分析に対する正式なアプローチを開発している場合、それらの分析の特徴を *TaskArchitect* のテンプレートに設定することができます。**Task Architect Inc** は御社にとって適切なスタイルを定義し、*TaskArchitect* ソフトウェアを貴方のニーズに合わせ、スタッフ訓練に役立てる際に貴方を支援します。詳しくは、[support@taskarchitect.com](mailto:support@taskarchitect.com) にお問合せください。

## 課題分析の概要

---

本書に記載され、*TaskArchitect* によってサポートされる種類の課題分析は階層的課題分析と呼ばれます。これは最も一般的に利用される種類の分析であり、訓練に要する時間が最小限で済みます。概念分析などの他の種類の課題分析ではユーザの課題についての洞察が得られますが、アプローチについてはより多くの知識が要求され、一般には学術分野で応用されています。

階層的課題分析では目標に向けてのユーザの行動を一連の課題に分解します。各課題については繰り返し再分析を行って課題を一連のサブ課題に分類したり、それらの課題を一定のレベルにまとめることができますので、分析者は設計プロセスにおいて次のステップのための情報を用意することができます。例えば、分析される課題の目標が「財務報告書の作成」の場合、サブ課題には「会計書類の審査」や「収益の予想」などが含まれるでしょう。分析者の目標に応じて、それらの課題をさらにサブ課題に分解することができます。「課題」及び「サブ課題」という用語は、ユーザの活動の記述としては同じものを指しており、両者の唯一の違いは、「親」と「子」という用語間の関係と全く同じように、分析の階層における相対的な位置だけです。「親課題」と「子課題」はしばしば課題とそのサブ課題を記述するのに使用されます。

特定の分析レベルでいつ停止するかは分析の「停止規則」と呼ばれます。これが特定の設計上の問題を解決するのに要求される最大限の詳細度について分析者の注意を喚起するので、課題の分析過剰による時間と労力の無駄が省かれます。例えば、ユーザの目標がマシンの主要な要素を明らかにすることである場合、停止規則は「マシンのどの重要な要素が使用されているかを明らかにする課題を再記述しないこと」であるかも知れません。分析者の中には設計上の特定の種類の問題のために正式な停止規則を定めている人もいます。以下の課題分析についてのより詳しい記載を参照してください。

課題の目標を達成するためにサブ課題が整理される方法は、サブ課題が実行されることについての簡潔で論理的な記述であるプランに記載されます。プランはサブ課題が実行されるべき順序と条件を記述します。例えば、ある課題に決まった順番で4つのボタンを点けることが係わる場合、プランの記述は次のようになるかもしれません。「1、2、3、4の課題を順番に行いなさい」。ここで各課題は1つのボタンの操作を記述します。

課題が記録されるにつれ、実際の課題記述に加え、係わってくるリスクや要求されるツールなどの追加情報を集めることもできます。これらは課題属性と呼ばれます。

*分析結果を表示するには多くの方法があります。次に挙げるのは最も一般的な方式です。*

- サブ課題（子課題や子）がその親課題とどう関連しているかを示す課題の字下げ項目リスト
- 課題同士の関係を示す階層ダイアグラム
- 各課題について記録される情報を示すテーブル
- 他のプログラムでの追加利用のため分析から編集される情報のリスト

*TaskArchitect* はこれらの出力スタイルのいくつかのバージョンに加え、他の分析ツールとのデータ交換を円滑にするための柔軟なデータエクスポート機能を提供します。

*次に挙げるのは課題分析で通常実施されるステップの流れです。*

- 何をどのように分析するかについて決定する。
- 情報を収集する。
- 収集された情報を検証する。
- 課題分析の表記法を適用する。
- 分析結果を表示する。
- 分析結果を使用する。
- 課題分析を更新管理する。

実施される分析の種類に応じて、課題について知られていることについてのブレンストーミングからユーザの代表的サンプルに対する詳細な聞き取り調査及び情報をクロスチェックするための2回目の聞き取り調査など、情報収集の内容も変わってきます。情報の記録はデータ収集の結果の単なる記述を超え、課題相互の関係と各課題についての属性の記録を表示するための情報表示における構造的なアプローチです。情報をこのように構造化することで分析における抽出漏れを見つけ出すことができます。例えば、分析者は課題について言及されていることに気づいても、十分な詳細は分からないので、データ収集段階に戻ることが要求される場合もあります。後の分析の段階では、情報の主要属性が導出され、プロジェクトにおいて後で利用する人たちに提供されます。出力内容は、課題の文章によるリスト、それらの関係を示すダイアグラム、全ての課題を文書化するテーブル、それらの関係及び重要な属性の頻度など様々です。

*TaskArchitect* は分析の各段階で役立ち、それらの全ての段階が一つのツールでサポートされているわけですから、必要に応じ各段階を繰り返し往復することも非常に容易です。例えば、課題のリストを *ListView* に素早くインポート又はタイプして、1回キーをクリックすれば階層ダイアグラムが出来上がり、後2回クリックすれば分析について報告書を生成できます。この統合により2つのキーストロークの利点が得られます。分析者は素早く最初の分析を行い、より大きく詳細な活動を対象にするかチェックするため関係者に結果を示すことができます。二番目に、分析者は入力と出力の段階を素早く往復できるので、アプローチを変更する必要があるかどうか確かめる前に膨大なデータ収集を実施する代わりに各段階を迅速に反復することができます。分析と生成される出力の種類に対するアプローチを迅速に変更できる機会があるので、課題分析に習熟するのに必要な時間が短縮されます。

## ステップ1：何をどう分析するのか

---

あらゆるプロジェクトに当てはまることですが、作業における重要なステップは分析の範囲を設定することです。考慮すべき要因には次の項目が含まれます。

### 分析の文脈

製品の使用状況分析では、オンにしてそれを使用し、またオフにするという主要操作に重点を置くことができます。しかし、製品がどのように取得され、設置され、アップグレードされるかを検討する際に関係者には幅広い価値が係わってくる場合があります。それは分析の本当のニーズが何かに左右されます。最初に、分析結果を利用することになる人々である関係者とこの点について追求することで、分析の価値を高め、作業の目標を見失わないようにします。

### 要求される課題詳細度

高いレベルでは会社に役割を提供するグループ全体の高い視点からのビューから、低いレベルでは特定のインタフェースで要求されるキーストロークとそれらのキーストロークを選択する背後にある認知プロセスなど、課題分析は非常に幅広い詳細度で行うことができます。分析は初めてという人たちは、問題の課題を本当に達成するために実際に何が必要とされているかについて時折考慮することなく、課題の詳細を捉えることばかりに多くの時間を費やす誘惑にかられます。安全性が最重要な全ての課題を明らかにするために包括的な分析が必要だろうか、それともシステムの性能を幅広く定義するために高い視野からの課題分析が必要だろうか。

### 記録する必要のある課題属性

目標を分析するために要求されるステップを単に記録するだけで設計上の問題が解決されることもよくありますが、活動の重要な側面を引き出すため課題についてもっと情報を記録することも時には有用です。これらの情報は課題属性と呼ばれます。例えば、安全性に関する追加の訓練を行うのが最も効果的な領域を決定するためには、課題ごとに安全上の重大なミスを犯すユーザの可能性についての実際又は予想される確率を記録することが分析にとって有用な場合があります。重大なミスが最も起こりやすい課題を記載するレポートは（ユーザマニュアルの第7章の「**レポートの作成**」を参照）、訓練プログラムを設計する際の指針になります。TaskArchitect を使うことでユーザは分析の開始時点（ユーザマニュアルの第2章の「**テンプレート**」を参照）又は途中（ユーザマニュアルの第4章の「**属性の作成**」を参照）のいずれでも記録されるべき膨大な課題属性を定義することができます。

分析を進めるにつれ、アプローチ又は記録される情報における詳細度を変更したい場合もあるはずで、TaskArchitect を使うことで、そうした変更がいつでも行えます。分析者としての経験が増すにつれ、問題へのアプローチにとって最も適切だと確信できるアプローチのスタイルをより簡単に選択できるようになります。

## ステップ2：情報の収集

---

誰が最も有益な情報を提供してくれるかを決定するため最初に投入される時間及び疑わしい結果を排除する心構えは、後の段階で大きな利益をもたらします。作業の初期の段階では、主なユーザや主題の専門家を明らかにし、次に経験についてのインタビューを通して課題には係わるが中心的ではなく、活動の一部についてだけしか係わらない他の人々の氏名を照合確認します。

既存の文書や訓練を調査することでインタビュー中の討論についての基礎が得られ、インタビューの対象となる異なるタイプの人々を識別するのに役立つ場合があります。

既存システムを分析している場合は、インタビューされる人の信頼を確保し、課題がどのように実施されるべきか又はマニュアルにどう書いてあるかではなく、課題が実際にどう実施されているかについて話してくれるようにすることが大切です。現在のシステムがユーザにとっての作業を困難にしている領域を明らかにする上でこの違いは極めて重要です。もちろん、収集されるデータの価値は分析対象の課題についてユーザがどれだけ知っているかに左右されます。

インタビューを組み立てる際にいくつかの基本的な質問戦略を応用することで、貴方にとっては情報の理解が容易になり、被面接者にとっては情報を提供しやすくなります。通常は異なる活動どうしのつながりを理解することが最初のステップになります。インタビューでは高い視点の課題から開始して、可能なら、概要が引き出された後でより詳細な部分に進んでいくべきです。もちろん、会話ですから通常はそう簡単にはいかず、インタビューがある特定の課題の些事に流れてしまうこともありがちです。

インタビューと貴方が発する質問の目的を明確に説明することで、会話の脱線を避けるのに役立ちます。問題領域の概要を理解することから始め、実施されている主な課題について質問します。次のような質問を使って各領域を順番にフォローしていきます。

「課題Xはどのように実施されていますか。」	「それをどうやるのか説明してくださいませんか。」
「Xの前はどうなるんですか。」	「Xの後はどうなるんですか。」
「そのために何を使用するんですか。」	「どういう情報を必要としていますか。」
「なぜその順番で行うんですか。」	「それが完了したってことはどうして分かるんですか。」
「それがうまく行かなかったらどうなるんですか。」	「何か問題が起こったらどうなるんですか。」

既に質問した課題とまだ質問していない課題を追跡できるようなメモの取り方を採用すると役立つでしょう。インタビューをうまくこなしていく方法は、それを録音するか、ペアで行うことです。通常は録音テープが便利ですが、実際に課題を実行している人々を記録するためビデオが使用されることもあります。2人組でインタビューするとき最も効果的な方法の一つは、一人を質問者に決め、もう一人を記録係にし、最初の方が質問切れになったときに役割を交替する方法です。これによって質問者がメモではなく聴くことに専念できます。

聞き取り対象者の協力を取り付けて課題階層を埋めるのを手伝わせるのでない限り、文書に情報をそのまま入力させるのは現実的なアプローチではありません。この場合、*TaskArchitect* のリストビューを画面上に表示することが、一人又はそれ以上の聞き取り対象者に情報を迅速に提供させ、途中で課題構造を修正させるための基本として使えます。全ての意見を聴いてあげて、相違点についての議論はまとめて後で対処できるように、グループ貢献の活力をうまく取り仕切ること十分な注意が払われる限り、このアプローチでは収集された情報の検証に後で要する作業量を削減することができます。

分析のための情報収集の際には、質問紙、観察調査及び重大な事件の調査も重要な役割を果たすことがあります。

### 停止規則の適用

停止規則は、調査対象の現在の問題を解決するために現在のレベル以上の課題の詳細な分析は必要ではないと決定する際に分析者が作業に適用する経験則です。

課題分析において使用される典型的な停止規則は、 $P \times C$  規則です。すなわち、 $P$  は課題における不適切な遂行の確率 (*probability*) についての予想であり、 $C$  は怪我や商業上の損失を含む不適切な遂行のコスト (*cost*) です。このリスクの推定は、システムの所有者によって定義されるシステムにおけるリスクの許容レベルを維持する必要性に応じてどの課題に追加の分析が必要かについての決定に適用されます。実務ではこれは目安としての規則ですが、顧客又は大きな設計チームがそれらを検討できるようになるまでの間は、分析者が決定を下す際に役立ちます。

これは停止規則の例ですが、停止規則の選択は分析の目的によって異なります。例えば、設計されるシステムの範囲を決めるためにそれが適用される場合は、最初は高いレベルでの分析が適切になるでしょう。停止規則は、「システムの範囲を理解するために十分なレベルの詳細が解明されたときに停止する」というものにもすることもできます。

### ステップ3：収集された情報を検証する

---

1 回のインタビューで活動に関する全ての情報を集め、最初からうまくやることを期待してはいけません。

当初の被質問者及び同じ活動を実施する他の人々について結果をクロスチェックすることで、貴方が集めた内容を検証し、抽出漏れを埋めるのに役立ちます。

書かれたメモやテープ録音されたインタビューを調べるのはとても時間がかかり間違いも生じやすいので、調査結果の要約を作成し、主題の専門家にそれを審査させるのが最も楽な場合もあります。これは課題分析に係わるステップ同士がどう重なり合うかのもう一つの例であり、次のステップである提示段階を実施しながらの方が検証の一部を一番うまく行うことができる場合が多いものです。

収集しながら結果を *TaskArchitect* に入力する利点の一つは、そこから提供される情報についての異なる見解によって、主題の専門家が結果を素早く評価して意見を出すことができることです。*TaskArchitect* を中心的なツールとして採用することで、主題の専門家は提示内容に素早く精通できるので、貴方と一緒に情報を充実させ検証することが容易になります。

## ステップ4：課題分析の表記法の応用

---

これはローデータからのメモと他の記録を課題、サブ課題、課題プラン及び課題属性に関する明確な記載に変えるプロセスです。このようにして集められる組織された情報は、課題階層と呼ばれます。課題を *TaskArchitect* に入力すると、階層的課題分析の表記法が適用され、自動的に更新されます。これは課題分析を手作業で実施することに比べ大きな利点です。

利用する階層の個々の様式は分析の目的及びどの段階まで作業が進んだかによって異なります。

**表示の主要部分は次のとおりです。**

- 課題の簡潔な記載 — 短く簡単なものにする。一般にサブ課題は親課題の下に列挙され、それらの階層関係を示すため字下げ処理されます。例えば一番上の階層は 1、2、3 で、次の階層は 1.1、1.2、1.3、さらに次は 1.1.a、1.1.b という具合に、階層を強調するために番号付与の規則が使用されるときもあります。
- 課題どうしがどのように調和するかについての短い記載 — 課題プラン。
- 課題について記録される関連属性 — 例えば、期間、成功の可能性、使用されるツール又はオペレーター、従属関係、頻度。

これらの観点については以下でさらに詳しく取り上げます。これらは貴方の分析の出発点であり、分析者の多くはこうした主要要素を表現する独自のスタイルを確立しています。

### 課題記述

課題記述は1つの目標に向けられた活動についての短い記述です。課題は貴方が記述する課題階層レベルで高いこともあり（自動車の運転）、その場合、それを完全に記述するための詳細な記載と追加のサブ課題が要求されることもあります。他方で、階層で低い方の課題（照明スイッチを点灯する）についてはそれ以上の記述が必要でないこともあります。課題記述は明快、完全、簡潔且つ実施される活動に関連したものでなければなりません。通常は動詞と目的語で構成され、修飾語が含まれることもあります（例：ボックス（目的語）を半分だけ（修飾語）開く（動詞））。

実際の記述では、設定済みの動詞のリストや標準課題に選択肢を制限するなど、貴方の組織によって定義される設定済みの書式に従うことになるかも知れません。

各上位レベルの課題（親）の下位にあるサブ課題（子）を含め、課題は実行される順序で列挙します。実行されたときにサブ課題の全てが親課題の目標を完全に達成し、サブ課題どうしが重複していないことを確認します。例えば、親課題が「タイヤを交換する」の場合、全てのサブ課題が完了したときに、タイヤが交換されていないと見なされればならず、サブ課題にはファンベルトの交換などの他の活動に関連するス

トップが含まれてはいけません。

ローデータを課題からさらにサブ課題に組織化する過程で上位レベルの課題の目標と全体的な目標を達成するため個々のステップがどう明確に調和するかが判明します。複雑な課題の分析では、分析者の課題は簡潔な課題記述と簡潔なプラン（以下を参照）及び小さく簡潔にまとめられたサブ課題のグループを作成することです。分析者はこれを行うことで課題の基本的な部分を明らかにし、システム全体を理解しやすいものにします。こうした理由から、課題については直接関連するサブ課題を7つを超えて作成しないようにすることが推奨されます。

*TaskArchitect* では課題ごとにサブ課題を最高で32まで追加することができます。しかしこの制限は例外的な事例に対処するためであり、推奨されるものではありません。*TaskArchitect* ではサブ課題を最高で20のレベルまで記録し、合計では4,000の課題を作成できるので、そこでの大きな制約はプログラムの能力というより課題を組織化の際の分析者の創造性ということになります。

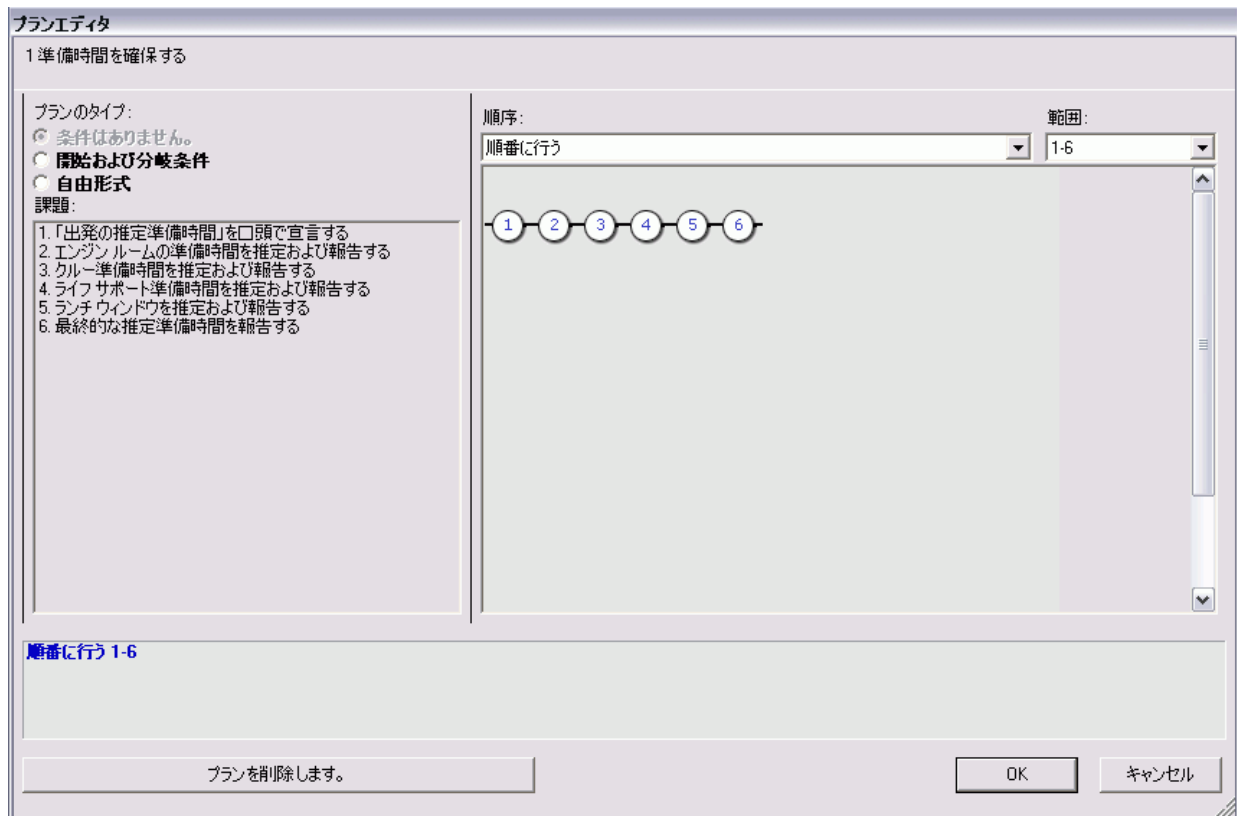
上記の制限を超える課題が要求される分析では、階層における各部をメインツリーから切り離し、別のファイルで作業することができます。この機能により分析者がツリーの各部を主要分析から分割することもできるので、他の分析者に作業を任せたり、後日に対処して、再度メインファイルに取り込むことができます。

## プラン

プランは親課題を達成するためサブ課題がどのように実行されるかについて記述するものです。プランではサブ課題が適用されるとき条件及びサブ課題が実行される順序を指定します。全てのサブ課題が特定の順序で1つずつ実施されるような状況など、プランの多くは非常に単純ですが、これらは複雑な課題が実行される方法を記述する際にも不可欠なものです。

*TaskArchitect* では課題を達成するためのプランをグラフィック形式で指定することができます。ユーザは課題が開始されるべき条件ならびに課題の目標を達成するために実施する必要のあるサブ課題のタイミングを指定することができます。*TaskArchitect* は課題の目標を達成するために様々な課題とサブ課題どうしがどのように調和するかを明瞭に示すダイアグラムを生成します。これによってプランの作成と評価が簡単になります。階層の中で課題が移動される際に、*TaskArchitect* はそれらの変更を反映させるためプランの内容を自動的に調整します。プランは *TaskArchitect* によって提供される分析のあらゆるビューに表示されます。

例えば、プラントの運転のためのプランを表示する場合、原材料がチェックされ、保守、アップグレード又は受渡しのいずれかの日常作業が実施される前にプラントが始動され、最後にプラントが停止されるといった手順になります。



**Figure 2 The Plan Editor.**

プランには通常、「x の場合は、y を行え」といった開始条件が含まれます。多くの課題の場合、「あらゆる場合に y を行え」といった 1 つの条件だけが存在します。TaskArchitect はこれらの選択箇所を条件の使用により定義することができます。条件は開始条件（プランの開始に至る）にしたり、分岐条件（プラン内部の条件）にすることができます。

次のように様々な種類のプランがあります。

#### 固定シーケンス：「順番に行う」

「1、2、そして次に 3 を行う」のように操作手順が固定されています。開始条件に応じて、特定の課題について全てのサブ課題が実施されるわけではないプランを作成することも可能です（例：「x の場合は、1、2、次に 4 を行いますが、y の場合は 1、2 そして 3 を行う」）。

#### 可変シーケンス：「いずれかの順序で全てを行う」

サブ課題はどの順序でも実行できますが、それら全てが実行されなければなりません。この例としては、実行リスト上の項目を確認する手順があります。

#### 並行操作：「同時に行う」

この条件ではリストされるサブ課題を同時に行うことができます。同時のサブ課題のあるこの種類の課題の例としては、「フロントデスクを担当する」、「書類業務を実行する」、「お客

を監視する」の手順が同時に行われる場合などが挙げられます。

**選択的完了：「任意のリストからいずれかを行う」**

メイン課題の目標を達成するためにサブ課題のいずれかを実施することができます。例えば、ドアを閉めるには、ドアを押すか、ドアの閉鎖ボタンを押す、又はドアに寄りかかるといった動作があります。

### 「...まで繰り返さない」という周期シーケンス

一定の条件が達成されるまでプランの中のサブ課題の全てが繰り返されるようにします。例えば、箱がいっぱいになるまで、製品の品質をチェックし、製品を箱に詰めていく場合等です。*TaskArchitect* で周期を表現する方法では、完了時の条件に応じて、「箱が一杯になるまで」、「製品のチェックを行い」、次に「製品を箱に詰める」といったプランを作成することになります。

### 選択肢：「この場合はあれをする」

これらの課題では、プラン内の決定地点に到達するまで、活動がプランに従って実行されます。例えば、製品を販売するため、顧客に挨拶し、顧客のタイプを判断し、顧客が見て回っている場合は、サポートします。代わりに顧客に買う意思がある場合、売込みを行います。*TaskArchitect* はプラン内で分岐条件を使用することでこの種のプランをサポートします。

### 条件シーケンス：「システムをモニターしてから行動する」

ここではシステムが特定の状態に達したときに課題が実行され、前の課題が完了したからという理由だけでは課題は実行されません。例えば、木材を加工しているときに大工さんは所定のレベルに達するまで計画行動を監視し、それから木材をサンドペーパーで磨く必要があります。やはりこの種類のプランを作成する際にも分岐条件が使用されます。

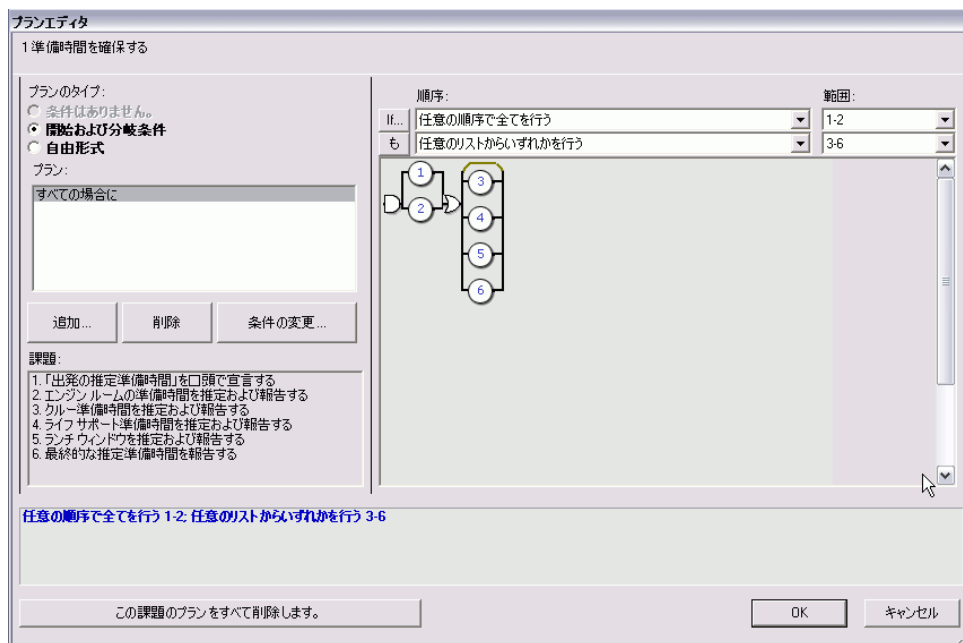


Figure 3 Branches within plans

短い課題記述を作成するのがときには困難に思える場合もあるし、明瞭で簡潔なプランを作成するのはさらに困難な場合もあります。これは、新たな課題を作成するか現在の課題階層を再配列することで

分析を変更する必要があるかも知れないという表れです。これには課題に関する概念の再構成が係わり、面倒だと感じることもあります。しかし、より単純な分析の方が最後には有用であるのが普通です。複雑な分析を解体する過程で課題を実行する現在の方法における混乱や問題を明らかにできるかも知れません。

## 課題属性

課題とサブ課題のリスト及びプランを決定することで活動全体の基本分析を行うと、プロジェクトにおける次のステップの十分な情報が得られるものです。しかしプロジェクトの中には、詳細な分析とより詳細な情報が要求されるものもあります。各課題についての情報の定義済み分類である課題属性の収集によってこうしたより詳細な分析が可能になります。

以下に挙げるのは課題属性の一部の例です。

- 課題を始動させる合図
- 精度、速度など、課題が実行されるべき基準
- 課題の成果
- 課題を実行するために使用されるツールと設備
- 遂行能力改善のために使用されるか要求される作業補助
- 使用されるマニュアル／使用されるスクリーン
- 良好な実績を確保するための監督活動
- 課題実行中の物理的要求
- 関連する環境条件
- 課題が実行される場所
- 状況－緊急時と通常の場合
- 要求される訓練の記述と量
- 課題の複雑性
- 要求される技能、知識及び経験
- 正しくない履行状況に伴う危険や潜在的な安全面での影響
- 正しくない履行状況に伴う潜在的な事業コスト
- ミスの確率
- 課題の困難さと頻度
- オペレーター
- 役割
- 情報源－属性情報の入手先

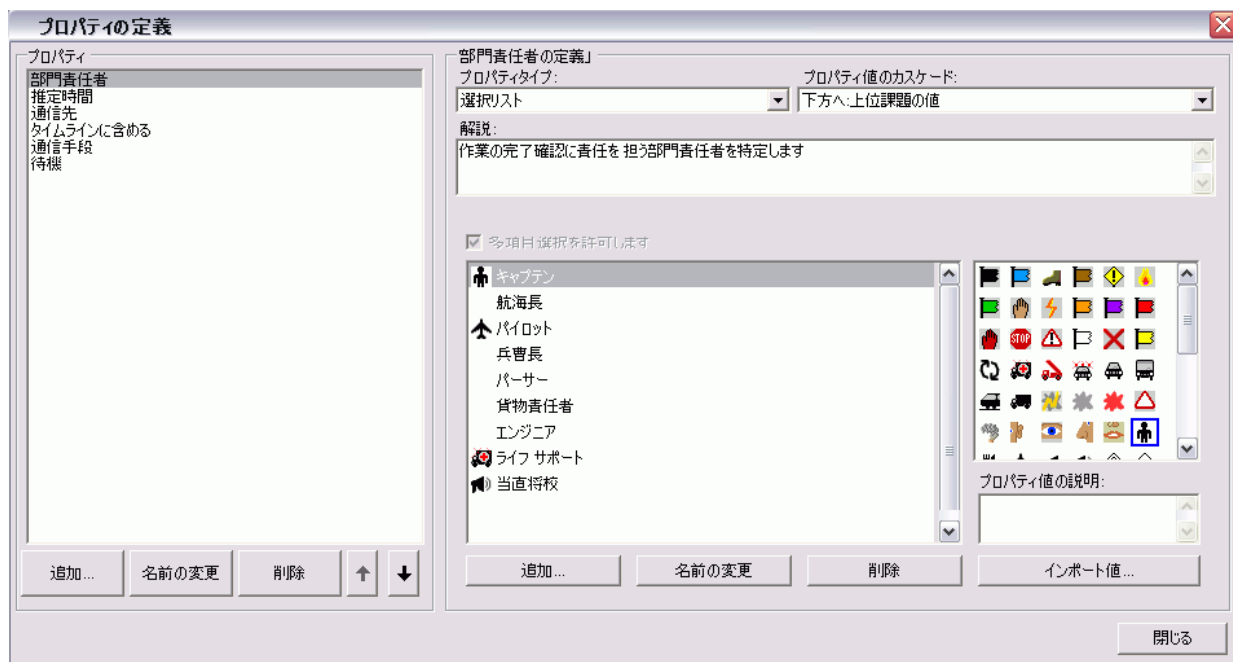


Figure 5 and 6 Defining properties for the analysis

属性はデータベースの参照を構築するのにも使用できます。例えば、データベースのレコードのキー（一意の番号など）を課題属性として記録することで、課題をデータベースの特定の項目にリンクさせたい場合もあるでしょう。

*TaskArchitect* では分析ごとに最高で 40 までの属性、多項選択式の回答のある属性については最高で 20 までの可能な回答を定義することができます。

*TaskArchitect* ではユーザがテンプレートを使って分析の開始時点でそれらを定義するか、分析を進める過程で Define Properties 機能を使って定義することができます。

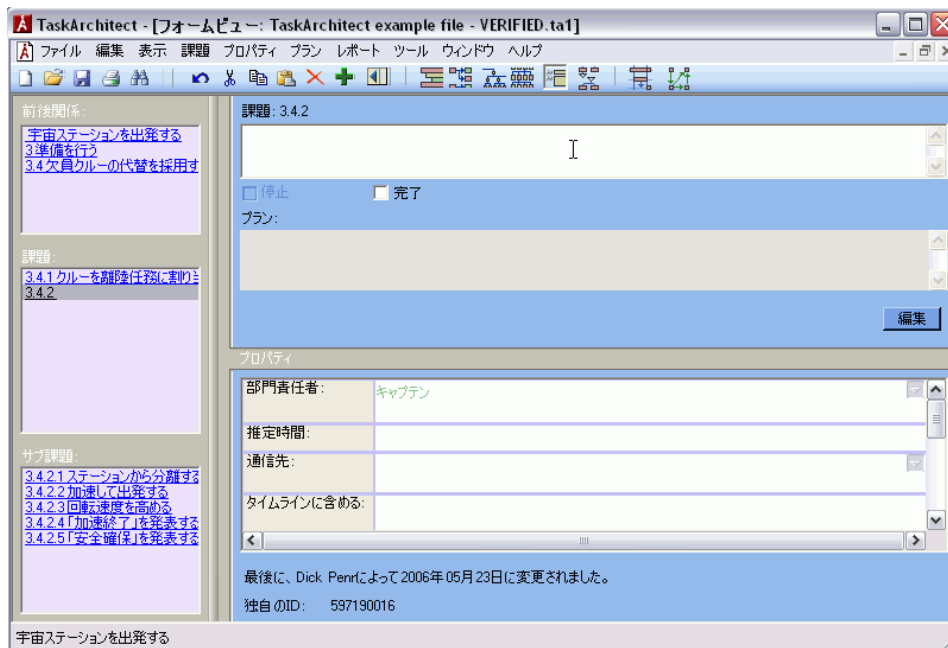


Figure 7 The Form View.

属性はフォームビュー又は Property のウィンドウ枠を使用して直接入力することができます。

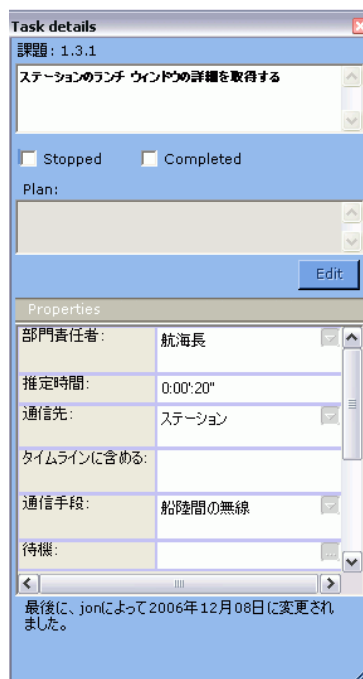
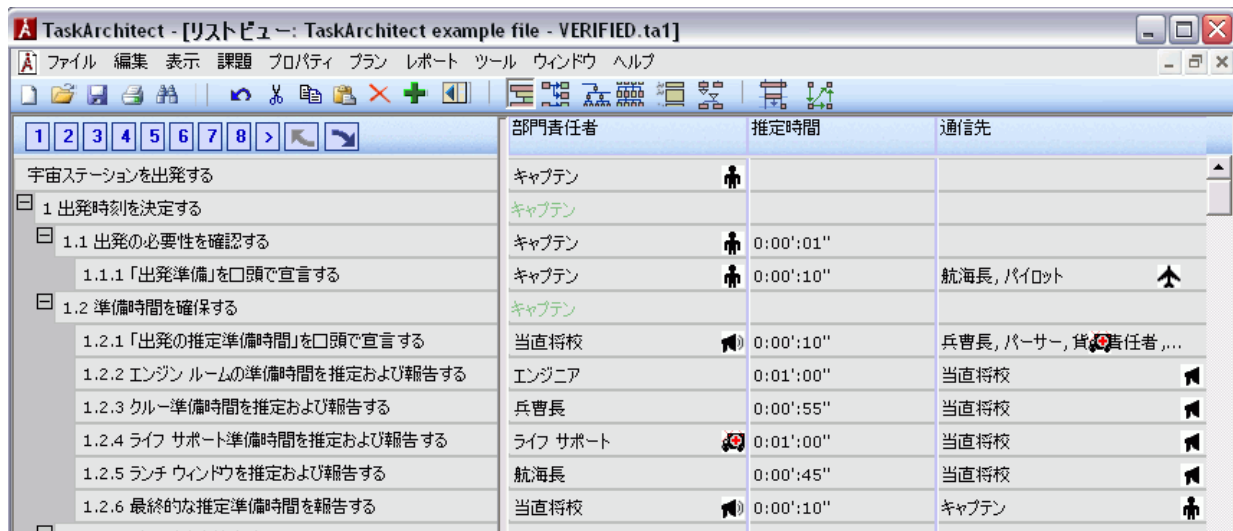


Figure 8 Task Details window

分析を進めるにつれ、属性について入力された値をリストビューに表示させたり、カスタムレポート

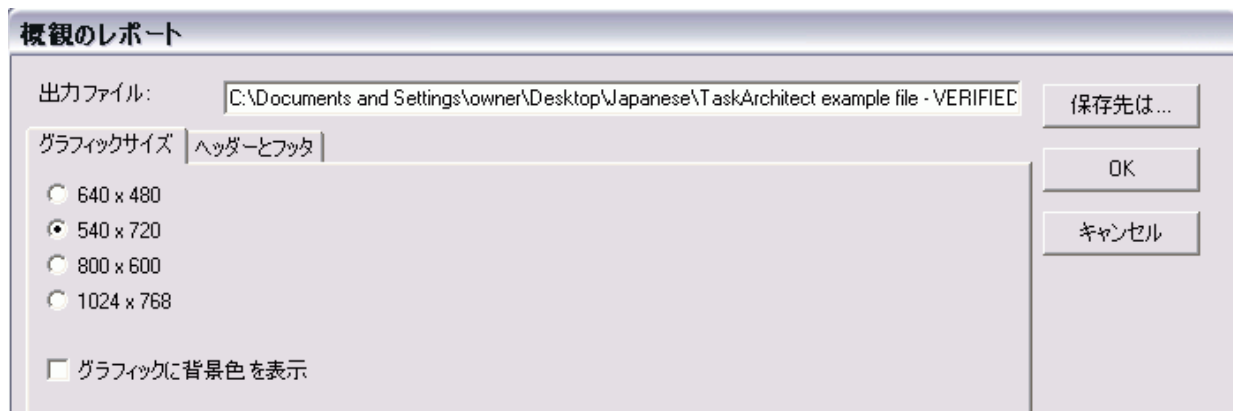
に出力させるか又はプログラムからエクスポートすることができます。



The screenshot shows the TaskArchitect application window with a list view of task property values. The window title is "TaskArchitect - [リストビュー: TaskArchitect example file - VERIFIED.ta1]". The menu bar includes "ファイル", "編集", "表示", "課題", "プロパティ", "プラン", "レポート", "ツール", "ウインドウ", and "ヘルプ". The toolbar contains various icons for file operations and task management. The list view has columns for "部門責任者", "推定時間", and "通信先".

	部門責任者	推定時間	通信先
宇宙ステーションを出発する	キャプテン		
1 出発時刻を決定する	キャプテン		
1.1 出発の必要性を確認する	キャプテン	0:00:01"	
1.1.1 「出発準備」を口頭で宣言する	キャプテン	0:00:10"	航海長, パイロット
1.2 準備時間を確保する	キャプテン		
1.2.1 「出発の推定準備時間」を口頭で宣言する	当直将校	0:00:10"	兵曹長, パーサー, 貨物責任者, ...
1.2.2 エンジン ルームの準備時間を推定および報告する	エンジニア	0:01:00"	当直将校
1.2.3 クルー準備時間を推定および報告する	兵曹長	0:00:55"	当直将校
1.2.4 ライフ サポート準備時間を推定および報告する	ライフ サポート	0:01:00"	当直将校
1.2.5 ランチ ウィンドウを推定および報告する	航海長	0:00:45"	当直将校
1.2.6 最終的な推定準備時間を報告する	当直将校	0:00:10"	キャプテン

Figure 9 List View showing property values



The screenshot shows the "概要のレポート" (Summary Report) dialog box. It has a title bar "概要のレポート". The "出力ファイル:" (Output File) field contains "C:\Documents and Settings\owner\Desktop\Japanese\TaskArchitect example file - VERIFIED". There are buttons for "保存先は..." (Save As...), "OK", and "キャンセル" (Cancel). Under "グラフィックサイズ" (Graphic Size), there are radio buttons for "640 x 480", "540 x 720" (selected), "800 x 600", and "1024 x 768". There is also a checkbox for "グラフィックに背景色を表示" (Show background color in graphics).

Figure 10 Creating reports

### 課題の番号処理

課題分析で最も時間のかかる作業の1つは、特定の課題やサブ課題を識別し、階層内に位置づけるために使用される番号である課題番号の追跡処理です。一部の分析者は階層内の課題のレベルを反映させるため番号付与についての規則を使用しています。例えば、一番上のレベルを一桁の数値にし、次のレベルを小数点一位で示し、その下を文字で示すといったやり方です。TaskArchitect では上位3つのレベルそれぞれに異なる規則を適用し（数字、文字又は小数点）、残りのレベルについては従うべき規則を指定することができます。

*TaskArchitect* では課題の番号処理とその修正が非常に楽に行えます。番号は課題階層における課題の位置を示すために割り当てられます。課題が追加、削除又は移動される過程で、番号付与が自動的に更新されます。

## ステップ5：分析結果の表示

*TaskArchitect* は課題分析を迅速且つ簡単に表示するための異なるオプションを提供します。

分析の初期の段階では、階層における位置に応じて字下げされた単純な課題リストが情報を取り込み表示する際の迅速な手段を提供します。*TaskArchitect* のリストビュー (ListView) がこのビューを提供する一方、課題プランと課題属性を明確に表示するために、課題とサブ課題に自動的に番号を割り振ります。

	部門責任者	推定時間	通信先
宇宙ステーションを出発する	キャプテン		
1 出発時刻を決定する	キャプテン		
1.1 出発の必要性を確認する	キャプテン		
1.1.1 「出発準備」を口頭で宣言する	キャプテン	0:00:10"	航海長, パイロット
1.2 準備時間を確保する	キャプテン		
1.2.1 「出発の推定準備時間」を口頭で宣言する	当直将校	0:00:10"	兵曹長, パーサー, 貨物責任者, ...
1.2.2 エンジン ルームの準備時間を推定および報告する	エンジニア	0:01:00"	当直将校
1.2.3 クルー準備時間を推定および報告する	兵曹長	0:00:55"	当直将校
1.2.4 ライフ サポート準備時間を推定および報告する	ライフ サポート	0:01:00"	当直将校
1.2.5 ランチ ウィンドウを推定および報告する	航海長	0:00:45"	当直将校
1.2.6 最終的な推定準備時間を報告する	当直将校	0:00:10"	キャプテン

Figure 12 and 13 The List View

課題記述の右側のカラムに各課題に関連する属性を表示することで、分析の初期に課題を表示する別の一般的な様式として課題テーブルが作成されます。

*TaskArchitect* はテーブル毎に1つの課題を表示するビューとしてフォームビュー (Form View) も提供します。分析者はこれを使用して課題に関する全ての情報を1つの場所に順序正しくまとめ、図やリスト表示で作業するよりも簡単に課題属性を編集できます。

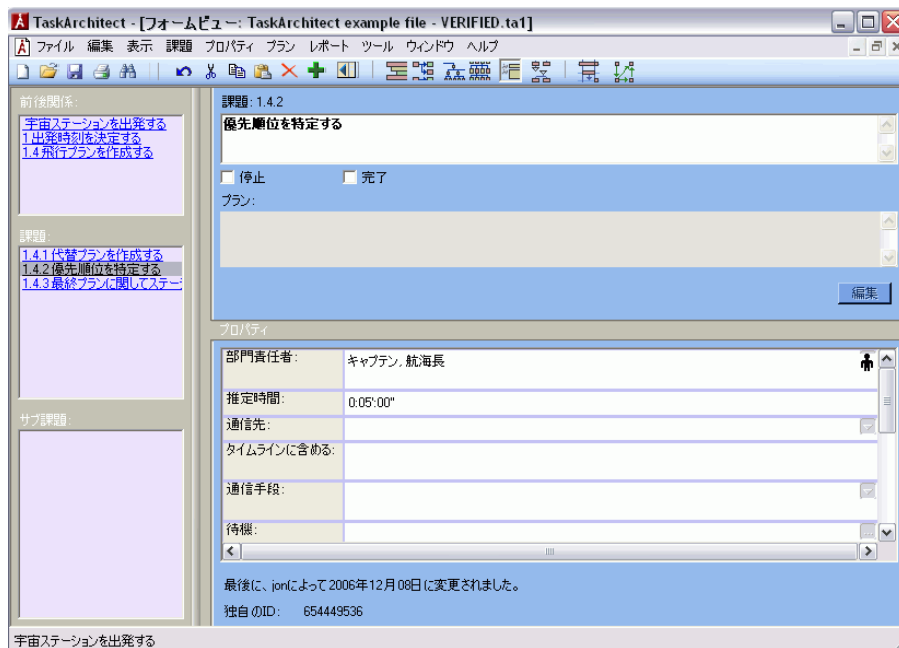


Figure 14 Form view showing all of the task information

課題階層を示すダイアグラムに含まれる情報はテーブルよりも詳細さに欠ける場合もありますが、検討者にとっては課題間の様々な関係を理解しやすくなります。順序から外れている課題がより簡単に際立ち、分析全体での課題のグループ分けと詳細度の分布がより鮮明になります。*TaskArchitect* はいくつかの形式のダイアグラムを提供します。

課題分析の結果を取り込むのに現在使用されるプログラムのほとんどは、結果をテキストによるリストか樹形図としてしか表示しません。そうした他のプログラムで樹形図を生成する場合でも、リストが変更されるたびに樹形図を編集し直す必要があります。*TaskArchitect* ではリスト、テーブル及びダイアグラムのビューを1回のクリックで切り替えられ、そのつど自動的に変更を組み入れることができます。

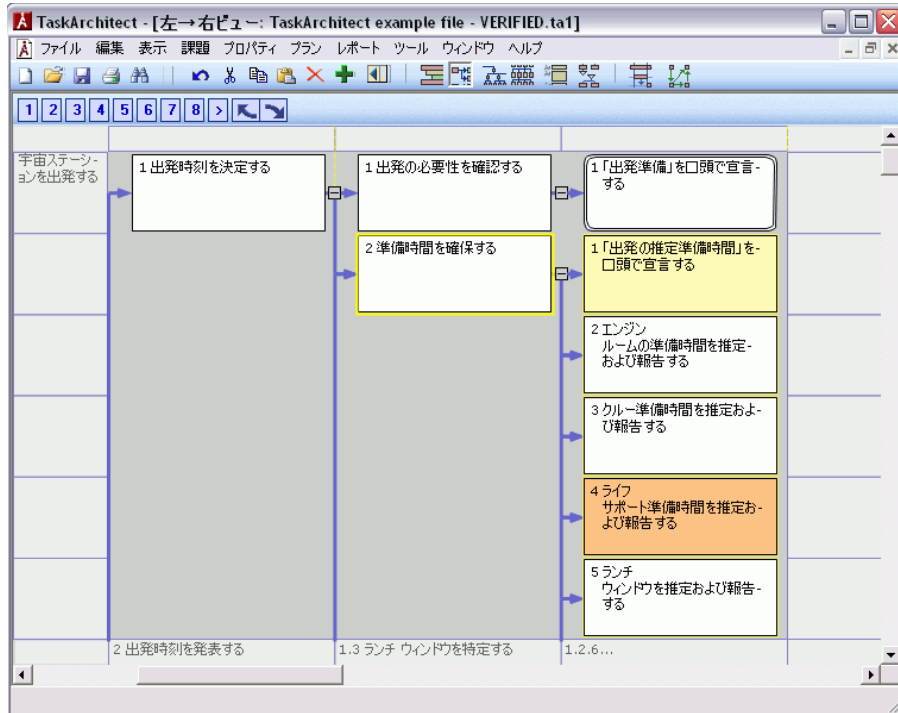


Figure 15 Left-to-right view of the tasks

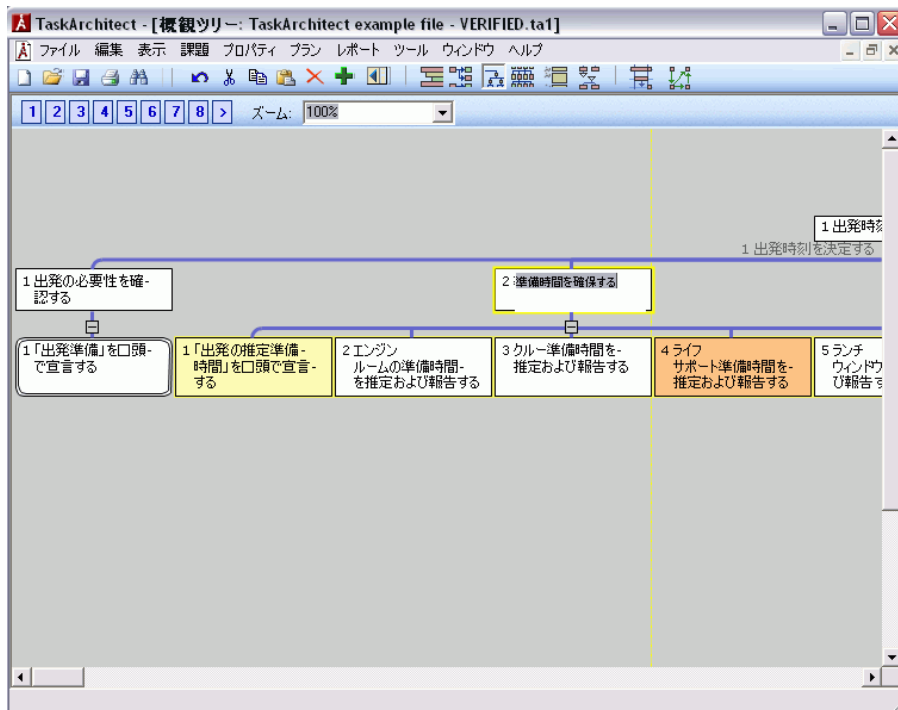


Figure 16 Overview of tasks

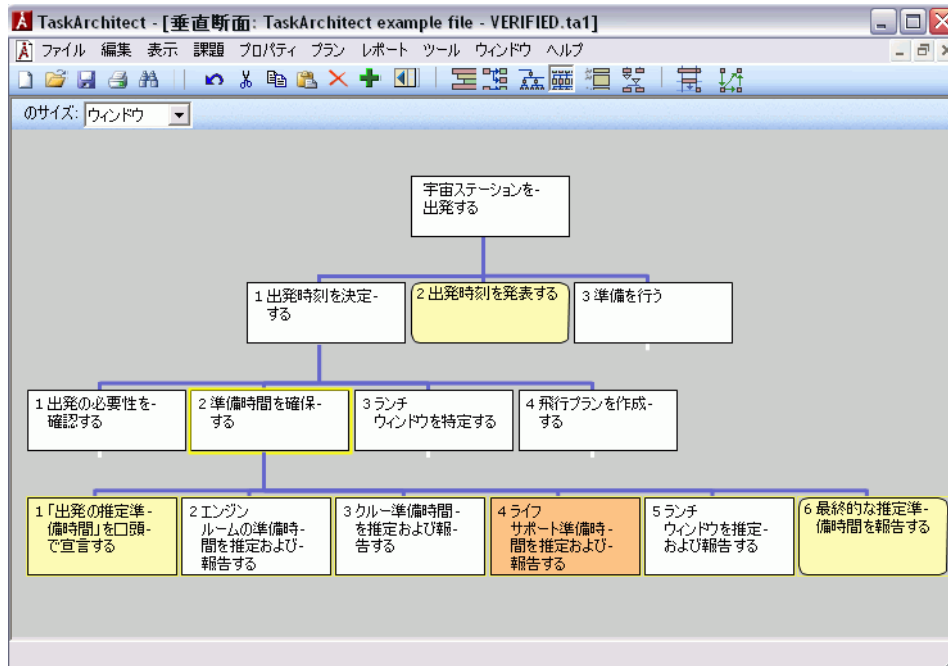


Figure 17 Vertical slice through the hierarchy

情報の適切な表示についての決定には、下す必要のある決定に関連のある情報は何か及びその情報をどう提示するのが最善かについての決定が係わります。このように結果の表示は結果の利用とも重なります。例えば、課題テーブルに示される属性を制限するか、特定の属性値のある課題だけを表示すると、提示内容を主な調査結果だけに絞ることになります。

*TaskArchitect* では課題をその属性値に従って分類し、レポート機能を使用してカスタマイズされた表を作成することができます。

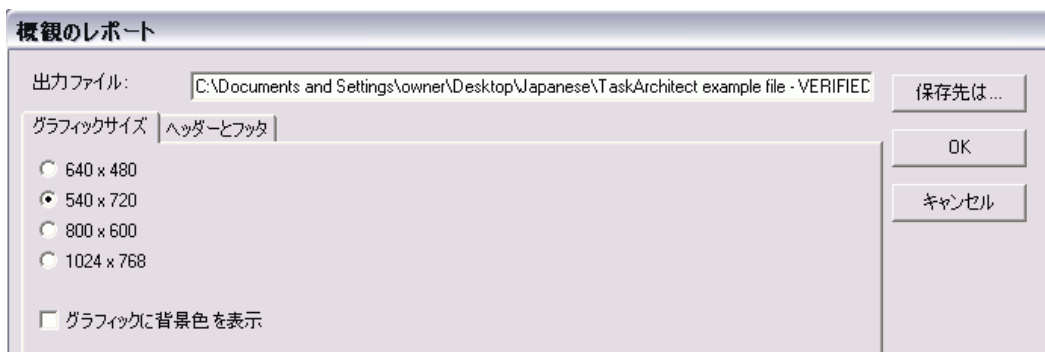


Figure 18 Generating Reports

結果を EXCEL 又は XML にエクスポートできる機能により分析者は専用ツールを使ってデータをさら

にソートしたり検討することができます。

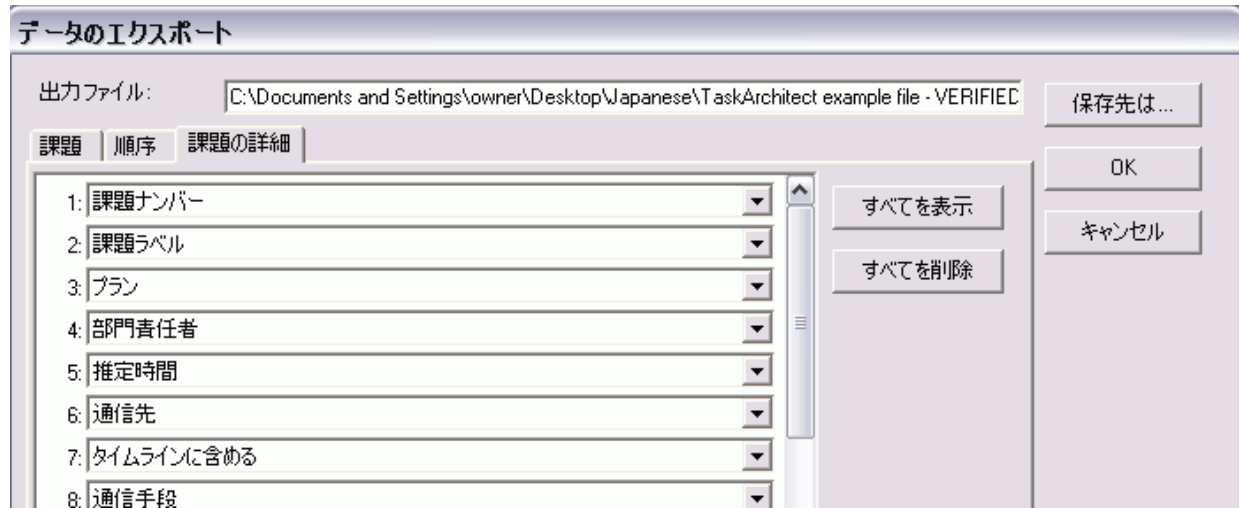


Figure 20 Exporting the analysis

## ステップ6：結果の利用

---

課題分析は設計上の健全な決定が行えるようにするため最適に提示される検証済みのデータを生成するのに使用されるツールです。ユーザの活動とその作業環境の表示は、ツール設計の改善、訓練の改善、選考、職場の設計などを介して彼らの遂行能力をどのように改善できるかについての仮説を生み出すのに使用されます。

前のステップ5では、さらなる検討のため最も重要な課題を選び出すため、課題分析をとおして収集される情報を課題属性に従ってどのようにソートできるかについて説明しました。例えば、訓練制度の設計者としては、最も頻繁に実行される課題でミスにつながる可能性が高いのに既存の訓練プログラムの対象になっていないものはどれかを知りたいはずです。

他のプロセスや製品の設計者の場合、大事なのは課題間の流れかも知れません。例えばユーザインタフェースの設計の際には、ダイアログが作業の自然な流れに従い、情報はそれが要求されるときに表示されるようにし、ユーザとマシンの間で課題が適切に配分されるようにすべきです。*TaskArchitect* は表形式と階層表示の両方で関連課題がどのようにグループ分けされるかを示すことでこうした分析に役立ちます。

課題の階層ダイアグラムはシステム設計者との優れたコミュニケーション手段となります。システムの表示はしばしばシステムに関する設計者の概念と直感的に適合することが多く、表示は注釈が付けやすく、ユーザの異なる活動部分がどう調和するかについての全体像をうまく表現してくれます。設計について合意ができれば、当然ながら次のステップは文書担当と訓練のチームがこの共通のシステム表示に基づきそれぞれの貢献を行います。

課題分析を使用するビジネス上の利点の例について詳しくは、[www.TaskArchitect.com](http://www.TaskArchitect.com) を参照してください。

## ステップ7：課題分析の更新管理

---

人間の処理能力を支援するための長期的プログラムでは、分析者はユーザの課題についての理解を改めていく必要があります。ツールは変わり、事業目標も変わり、労働者すら変わります。定期的見直しによって分析を最新のものに維持し、組織にとって有益なものにします。

## 課題分析の種類

---

*TaskArchitect* は「課題分析」の名の下で実行できる広範な活動をサポートするように設計されています。ツールを途中で設定することができ、分析者は自分にとって重要な情報を完全に取り込むため課題属性のリストを調整することができ、広範な種類の表示と出力の機能が用意されているので、分析者がツールに使われるのではなく、ツールを使いこなせます。*TaskArchitect* をどのように利用できるかについての着想を刺激するため、次の節ではこのプログラムでもサポートすることのできる、職務分析と認知的課題分析の2つの種類の課題分析を取り上げます。

### 職務分析

職務分析はシステム設計よりも訓練及び人事の活動により関連性があることがしばしばです。ここでは、ある者が実際に何を行うかについての詳細というより、職務に関連する活動の種類（人間関係、経験、知識、技能、職務上の責務）を明らかにすることに重点が置かれます。課題の分析と同じように、分析ではインタビューの対象者ではなく、職務の要件と組織内での職務の役割に重点が置かれます。

### 利点

職務分析は次の領域での決定が行えるようにするため、職務に関する基本的な情報を提供するツールです。

- 訓練制度の開発
- 人事の選考、評価、昇進
- 職務記述書と職務分類の作成
- キャリア管理
- 報酬
- 職務設計
- 安全性評価
- 既存能力と要求能力の相違
- 既存の訓練、選考、報酬及び業績評価方法の適合性の評価
- 訓練効果の測定
- データ収集

職務分析は配属人数など職務についての基礎的な情報を収集するための質問紙から開始することが多く、次のような項目が対象になります。

- 組織における当該職務の目的
- 職務内容：頻度、期間、労力、複雑さ、基準、課題間の関係など。これは前に取り上げた課題分析です。
- 職場の人間関係：監督者と被監督者、内部及び外部の人間との関係
- 職務を遂行するために要求される知識、技能及び能力（KSA）
- 防護服を含め使用される機材
- 労働環境：物理的条件、職務の社会的条件（単独作業、他人との協働、納期に基づく作業など）、危険、勤務の場所及び通勤の必要性

この後には、矛盾を解明し、欠落を埋めるための個人面接及び上司との資料の検討が続きます。

*他のデータ源には次の項目が含まれます。*

- 職務分類制度の見直し
- これまでの職務記述書、関連する職務記述書
- 作業員の観察
- 時系列データの検討

## *出力*

*TaskArchitect* の課題リスト、属性テーブル及び課題相互の関係のダイアグラムを使用して、収集された情報を照合し、提示することができます。職務分析の出力には次の項目が含まれます。

- 職務の記述と職務を遂行する典型的な個人
- 職務の課題リスト
- 順調な職務遂行に要求される知識、技能、能力その他の特性のリスト
- プロジェクトにおける次のステップの詳細を提供する職務の属性

## 認知的課題分析

このタイプの課題分析では課題遂行の基本にある認知プロセスを明らかにすることに重点が置かれます。職務遂行の成果が肉体的能力よりも思考能力に依存する課題に対処するときはこの種の分析が特に有益です。この分析には非常に時間がかかる場合があるので、かなりの意思決定や絶えず変化する状況での曖昧さや迅速な行動への対処が係わるような、観察できる行動の課題分析だけでは複雑な課題を理解するための十分な情報が得られない場合にだけ通常は利用されます。

これらの認知ステップ、つまり人間の頭の中で行われていることは観察はできないので行動から推察する必要があります。認知課題分析は人々の問題への取り組み方、知っている内容及び遂行能力の改善にその知識を生かすため知識をどのように組織化しているかについて理解することを目指します。

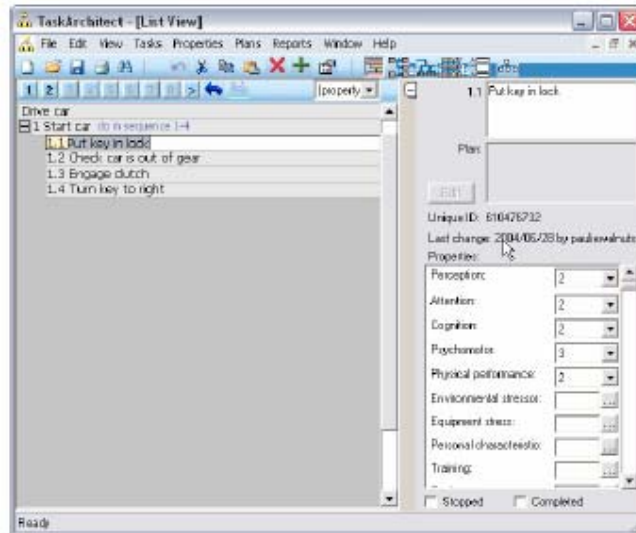
ここでも、係わってくる課題の分析がこの手の分析の第一ステップになります。そこから、基本にある関連する認知プロセスについての情報が導出され、記録されます。これには効果的な認識能力に寄与する要因、熟練者と初心者の行動の違い、認識能力の改善方法についての仮説などが含まれることもあります。

認知課題分析が使用されてきた分野には、仕事量の測定、異なるコンピュータインタフェース設計の仕事量への影響についての比較、絶えず変化する環境での迅速な問題解決と効果的な職務遂行のための訓練制度の設計などがあります。

分析における最初のステップは階層課題分析を使用する課題の分類です。そこから先のステップはプロジェクトの目標と分析者のスタイルによって異なります。複雑な課題における認知の分析については数多くのアプローチが開発されてきていますが、それぞれのアプローチは認知に関する分析者の仮説と解決しようと取り組んでいる問題の種類から生まれています。本書では認知課題分析の利点についてのあらましを述べていますが、参考文献の箇所はこの手法を実施する方法の手引きとなるものです。

データ収集の大半は、問題解決のための重要な手がかりの識別方法、問題の分類と優先順位の付け方、状況の見分け方など、異なる状況での問題へのアプローチに関する主題の専門家に対する詳細な聞き取り調査によって実施されます。貴方がインタビューを試みる専門家はその活動を「自動的に」行っており、ステップを明確にするのに困難を覚えるということが往々にしてあります。非専門家に話をするときにはステップを詳細に説明するのではなく、情報を簡略化する際に専門家を利用できる場合もあります。

アプローチの中にはユーザの遂行能力のモデル作成に基づくものがあります。課題の各ステップに要求される認知の種類ごとに評点が与えられます。課題が再配列されるか、異なるツールによって支援される際に、課題の総合的な困難さが再計算されます。最初の課題分析及び各課題に適用されるべき認知の重み付けを取り込むのに、*TaskArchitect* を使用することができます。



**Figure 21 Cognitive Task Analysis example.**

< 参考図書 >

Kirwan, B. and Ainsworth, L.K. (Eds.) (1992) *A guide to task analysis*. Taylor and Francis, London and New York

Jonassen D.H., Tessmer, M., Hannum, W.H. (1999) *Task Analysis Methods for Instructional Design*, Lawrence Erlbaum Associates, New Jersey

Shepherd, A. (2001), *Hierarchical Task Analysis*. Taylor and Francis, London and New York

Stanton, N.A., Diaper, D. (2003), *Handbook of Task Analysis for Human-Computer Interaction*, Lawrence Erlbaum Assoc Inc

Annett, J. (2003) Hierarchical Task Analysis, In Holnagel, E. (2003), *Handbook of Cognitive Task Design*, Chapter 2, pp17-35. Mahwah NJ: Lawrence Erlbaum.