特集 要求工学

4　非機能要求とゴール指向要求定義

山本 修一郎
（株）NTT データ技術開発本部システム科学研究所

最近の要求工学に関する国際会議（RE06，RE07，REFSQ06，REFSQ07）論文の約 35%が非機能要求とゴール指向に関する論文であることから分かるように，非機能要求とゴール指向は現代の要求工学の中で最も活発な研究分野となった．

本稿では，非機能要求（NFR，Non Functional Requirements）と主要なゴール指向要求定義手法について解説するとともに，ゴール指向と NFR や従来手法との関係についても紹介する．

非機能要求

非機能要求という概念には次に見るように異なる解釈があるので，どのような意味で非機能要求という言葉を用いているのか注意する必要がある．

まずソフトウェアの内部と外部，機能とその特性に分けて要求を考えると次のようになる（**図 -1**）．ここでソフトウェアの外部には人間による業務活動やデバイス，外部システムなどがある．

ソフトウェア内部の機能に対する要求が機能要求である．ソフトウェア内部の機能の性質に対する要求が機能属性要求である．ソフトウェア外部の機能に対する要求が外部活動要求である．ソフトウェア外部の活動の性質に対する要求が外部活動属性要求である．非機能要求の狭義の定義は機能属性要求である．これに対して非機能要求の広義の定義は，機能属性要求，外部活動要求，外部活動属性要求の総称となる．

ソフトウェア内部だけについて「要求」を用いて，外部については「制約」とする場合もある．いずれにしろソフトウェアを取り巻く外部との境界を明確にすることが重要である．

■ IEEE std 830-1998

IEEE が推奨しているソフトウェア要求仕様（SRS，Software Requirements Specifications）の記述法で扱われている非機能要求を図 -1 の整理に従って紹介すると**表 -1** のようになる．IEEE std 830-1998 には 4 章「良い SRS を作成するための留意点」と 5 章「SRS の構成要素」の中で，非機能要求に関する記述があるがここでは 5 章について説明する．なお表中の数字は 5 章の章節番号である．以下では，機能属性，外部活動，外部活動属性に分けて，どのような非機能要求を記述することが推奨されているのか説明しよう．

機能属性に関する非機能要求には以下の 6 種類がある．

①狙い

SRS の狙いと想定する読者を記述する．

②スコープ

ソフトウェアを開発することによって得られる利益や，ソフトウェア開発の目的（Objectives）とゴールを記述する．

③実現時期の配分

機能要求の提供時期を次期開発まで延期できるかどう

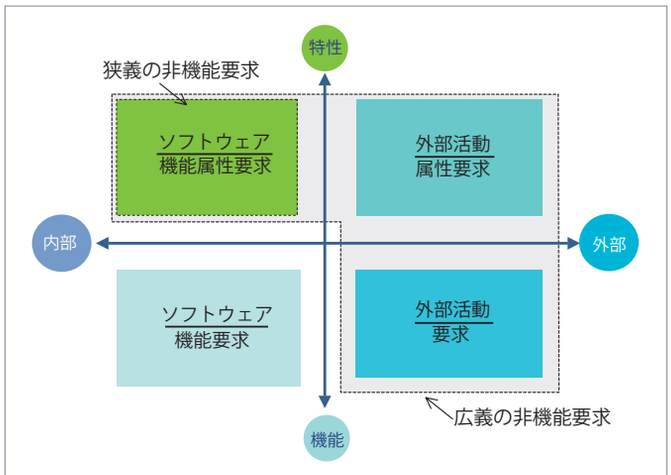

図 -1　機能要求と非機能要求

分類	SRS の構成要素	
機能属性	狙い，スコープ（利益，目的，目標）	5.1
	実現時期の配分	5.2.6
	性能	5.3.3
	論理データベース（使用頻度，アクセス性能，一貫性制約，保持条件）	5.3.4
	ソフトウェア属性（信頼性，可用性，セキュリティ，保守性，移植性）	5.3.6
外部活動	製品の位置付け（システムインタフェース，ユーザインタフェース，ハードウェアインタフェース，ソフトウェアインタフェース，通信インタフェース，メモリ，運用，サイト適応要求）	5.2.1
	外部インタフェース	5.3.1
	制約（制度，ハード制限，他 AP インタフェース，並列操作，監査機能，制御機能）	5.2.4
	標準化（書式，命名規約，会計手続き，監査証跡）	5.3.5
	付録（コード，媒体の梱包に関する指示事項）	5.4.2d
外部活動属性	ユーザ特性（教育レベル，経験，専門技能）とその必要性の根拠	5.2.3
	要求に重要な影響を与える前提条件，依存関係	5.2.5
	制約（上位言語，信号プロトコル，信頼性，臨界性，安全性，セキュリティ）	5.2.4
	付録（費用分析，背景情報，解決すべき課題）	5.4.2abc

表 -1　IEEE std 830-1998 における非機能要求

かを記述する．

④性能

ソフトウェア性能について静的数値要求と動的数値要求とを記述する．静的数値要求には，端末数，同時接続利用者数，処理対象とする情報量と種別などがある．動的数値要求には，正常時やピーク負荷時に対して一定時間内に処理可能なトランザクション数，タスク数，処理データ量などがある．

⑤論理データベース

データベースに関連する非機能要求には，使用頻度，アクセス性能，一貫性制約，保持条件などがある．

⑥ソフトウェア属性

- 信頼性：サービス提供時に必要とされる信頼性を記述する．
- 可用性：同期点，リカバリ条件，再起動条件などシステムに対して要求される可用性の条件を記述する．
- セキュリティ：不慮や悪意によるアクセス，使用，変更，破壊，情報漏洩からソフトウェアを保護する要因を記述する．たとえば使用する暗号技術，保存対象ログやデータ履歴，異なるモジュールへの指定された機能の割当て，プログラムの特定領域間での通信制限，重要データの一貫性検査などがある．
- 保守性：モジュール性などソフトウェアの保守を容易化するための属性を記述する．
- 移植性：他のホストマシンや OS などへのソフトウェアの移植性を容易化するために，ハードウェア独立性や移植性の高い言語の利用などについて記述する．

外部活動に関する非機能要求には以下の 7 種類がある．

① 製品の全体像

ソフトウェア製品が次のような制約を内部でどのように扱うかを記述する．

- システムインタフェース：システム要求を実現するソフトウェア機能を識別する．
- ユーザインタフェース：画面構成やメニュー内容などに関する論理的な特性ならびにユーザ特性に応じたインタフェースの最適化方法を記述する．
- ハードウェアインタフェース：ポート数や命令セットなどの構成上の特性やどんなデバイスをどのように提供するかなどの論理的な特性を記述する．
- ソフトウェアインタフェース：開発対象ソフトウェアが必要とするソフトウェア製品と，他のアプリケーションと開発対象とのインタフェースに対して，なぜそれらを必要とするのかという目的を記述する．
- 通信インタフェース：LAN プロトコルなどの各種通信インタフェースを記述する．
- メモリ：主記憶と補助記憶について特徴と制限を記述する．
- 運用：運用形態，運用時間帯やバックアップ，障害対応などを記述する．
- サイト適応要求：初期設定やインストールに関する制約を記述する．

②外部インタフェース

外部インタフェースについての非機能要求には，目的，データの有効範囲，精度，許容誤差，測定単位，タイミングなどがある．製品の全体像でも外部インタフェースを記述するので内容の重複を避ける必要がある．

③制約

制度，ハード制限，他 AP インタフェース，並列操作，監査機能，制御機能について記述する．

④標準化

ソフトウェア開発で従うべき書式，命名規約，会計手続き，監査証跡などについて記述する．

⑤付録

開発対象ソフトウェアの機密保護や輸出などに関連してコード，媒体の梱包に関する指示事項を付録で記述する．

⑥オブジェクト

システム内部で実世界の外部活動実体に対応するのがオブジェクトである．オブジェクトは属性と属性を操作する機能を持つ．

⑦フィーチャ

望ましい結果をもたらすために一連の入力に基づいてシステムが提供する外部サービスがフィーチャである．フィーチャは刺激と反応からなる対の系列で記述される．電話サービスはフィーチャの例である．

ここで，最後の 2 つの 5.3.7.3 オブジェクトと 5.3.7.4 フィーチャは IEEE std 830-1998 では機能要求を構成するための切り口を示す要素として記述されており，機能の外側にある概念を示していると考えられる．したがってこれらの切り口がなぜ，機能と関係するのか，なぜ必要なのかということを記述することは広義の非機能要求と考えられる．たとえば，i* でも，サービスやアクタを記述しておりこれらは機能要求の外側にあって，機能の存在根拠を示す重要な要素である．

外部活動属性に関する非機能要求には以下の 6 種類がある．

①ユーザ特性

ユーザには教育レベル，経験，専門技能などの特性がある．またユーザ特性を必要とする根拠も記述する．

②要求に重要な影響を与える前提条件

たとえば，想定したソフトウェア製品の実行環境が用意できなければ，要求変更が発生する可能性がある．

③制約

開発者の選択肢を制限する項目には，上位言語，信号プロトコル，信頼性，アプリケーションの臨界性，安全性やセキュリティについての考慮などがある．

④付録

開発対象ソフトウェアについての費用分析，背景情報，解決すべき課題などを付録で記述する．

⑤システムモード

正常時，緊急時，訓練時などのシステムの運用モードに応じて必要とされるソフトウェアの機能が異なる場合，システムモードを明らかにした上で対応する機能の目的や特性を記述しておく必要がある．

⑥ユーザクラス

顧客，保守担当者，営業担当者などの異なるユーザ種別ごとに必要とされるソフトウェアの機能が異なる場合，ユーザクラスを明らかにした上で対応する機能の目的や特性を記述する必要がある．

ここでも，外部活動の場合と同じように最後の 2 つの 5.3.7.1 システムモードと 5.3.7.2 ユーザクラスは IEEE std 830-1998 では機能要求を構成するための切り口を示す要素として記述されており，機能の外側にある概念を示しているので広義の非機能要求と考えられる．

以上述べたように IEEE std 830-1998 を参考にしながら，プロジェクトごとにどのように非機能要求を記述するかについて個別に判断する必要がある．たとえば，IEEE std 830-1998 では信頼性などのソフトウェア属性と性能とが独立した章立てになっているが，次に見る ISO9126 のように同列で記述する方法もある．

■ ISO9126

ソフトウェア品質を測定するための世界標準として定められた ISO9126 では，次の 6 項目の品質属性とその副特性を定めている．

【機能性】明示的および暗示的必要性に合致する機能を提供する特性

【信頼性】指定された達成水準を維持する特性

【使用性】理解，習得，利用でき，利用者にとって魅力的である特性

【効率性】使用する資源の量に対比して適切な性能を提供する特性

【保守性】修正のしやすさに関する特性

【移植性】ある環境から他の環境に移すための特性

また適合性（compliance）がすべての特性に共通する副特性となっている．

このように ISO9126 では機能に関する属性要求としての非機能要求を定義している．

■ 非機能要求の重要性

従来は，機能要求だけを対象として仕様化することが多かった．この背景には，機能仕様が明確に定義できていればソフトウェア開発が成功すると考えられていたためである．しかし，ソフトウェアがなぜ開発されるのかという目的や，なぜその要求が必要なのかという理由を明確にしておかなければ，せっかく開発した機能が使われないまま廃棄されるという無駄が発生する可能性がある．またどのような特性を持つ機能なのかが明確に定義されていなければ，性能や操作性の点で満足できないソフトウェアが生産され，大きな手戻りを生じることに

なる．

したがってシステム開発の目的や機能属性などを非機能要求として明確に定義することにより，これらの問題がソフトウェア開発で発生しないように備えることが重要になる．

ゴール指向手法

代表的なゴール指向要求定義手法として，KAOS[1]，i* フレームワーク[2]，NFR フレームワーク[3]などがある．本稿ではこれらの手法におけるゴールの定義を紹介する．ゴールの一般的な解説は文献 4），5）などにあるので参考にしていただきたい．

■ KAOS

KAOS にはゴールモデル，オブジェクトモデル，エージェントの責任モデル，操作モデルという 4 つのモデルがある．KAOS のゴールモデルでは，システムを構成するエージェントが協働することで実現される意図としてのゴールと期待や問題領域の制約などとの関係を記述する．ゴールモデルに基づいて責任モデル，操作モデル，オブジェクトモデルを作成する．責任モデルでは，エージェントとエージェントが責任を持つ機能要求との関係を記述する．操作モデルでは機能要求を実現する操作とそれを引き起こすイベントならびに，操作の入出力となるデータ実体を記述する．オブジェクトモデルでは，データ実体とその関係を記述する．

エージェントは，人間，デバイス，ソフトウェアシステムの構成要素である．KAOS では，あるゴールを実現するためにエージェントごとに役割が与えられ，ソフトウェアエージェントと環境エージェントの 2 種類に分けている．環境エージェントはソフトウェアエージェントが活動する上でのコンテクストを与えることになる．KAOS のゴールには提供すべきサービスを示す機能ゴールとサービスの品質を示す非機能ゴールがある．これに対して，KAOS では物理規則，組織規約，ポリシーなどをゴールとは区別して，環境に関する領域特性（domain property）であるとする．

KAOS では，AND 関係と OR 関係によって上位ゴールを下位ゴールに分解するゴールグラフを AND/OR 詳細化による抽象階層（refinement abstraction hierarchies）と呼ぶ．AND 関係では，上位ゴールを複数のサブゴールと領域特性によって分解することによって，すべての下位ゴールと領域特性が上位ゴールを実現する上での十分条件になることを示す．OR 関係では上位ゴールを詳細化する代替案の集合としての下位ゴールに分解する．

上位ゴールになればなるほど，抽象性が高くそれを実現するためには多数のエージェントの協働が必要になるというわけである．このようなゴール分解の終了基準は，すべての下位ゴールに対してそれを実現するエージェントが特定されることである．このとき，エージェントはその下位ゴールの実現条件を観測して，その条件が成立することを判定する責務がある．ゴールグラフで下位ゴールを持たないゴールを終端ゴールと呼ぶと，終端ゴールに責務を持つエージェントが特定できるまで分解を続けることになる．

開発対象ソフトウェアを構成するエージェントが責務を持つ終端ゴールを要求に対応付けている．これに対して環境エージェントが責務を持つ終端ゴールでは，ソフトウェアが環境に対して期待することを表現する．

また KAOS では，時間的な振舞いに関するゴールの種類を，生成的振舞いである達成／中断（achieve/cease）と，制約的振舞いである維持／回避（maintain/avoid）に分類することで，時相論理式で表現できる．たとえば**図 -2** に示したエレベータ制御のゴールグラフにある「エレベータの箱が移動中は扉を閉めておく」という達成型ゴールを次のような時相論理式で表現する．

$$\forall \text{ bx: Box} : \text{bx.Moving} \Rightarrow \text{bx.doorState} = \text{'closed'}$$

なお，図 -2 では，エレベータ制御の一部だけを記述している．たとえば，昇降ボタンなどを記述していない．

■ i* フレームワーク

i* フレームワークは，アクタ，ゴール，タスク，ソフトゴール，資源という 5 つの要素を用いて，現状のビジネスを理解したり，情報システム導入による効果などをモデル化して分析する手法である．i* の i と * はそれぞれ意図（intention）とネットワークを意味しており，i* は意図のネットワークという意味である．i* フレームワークのモデルには，アクタ間の依存関係を分析する SD（Strategic Dependency）モデルと，アクタ内部の依存関係を分析する SR（Strategic Rationale）モデルの 2 つがある．

SD モデルでは，ゴール，タスク，ソフトゴール，資源という依存内容（dependum）を提供するアクタ（dependee）とそれを期待するアクタ（depender）に関する 2 者関係を記述する．

SR モデルでは，タスク分解，手段目的分解，貢献関係を用いて，アクタ内部でゴール，タスク，ソフトゴール，資源を階層的に分解していく．タスク分解では，アクタが達成すべきタスクを下位ゴール，下位タスク，下位ソフトゴール，下位資源に分解する．手段目的分解で

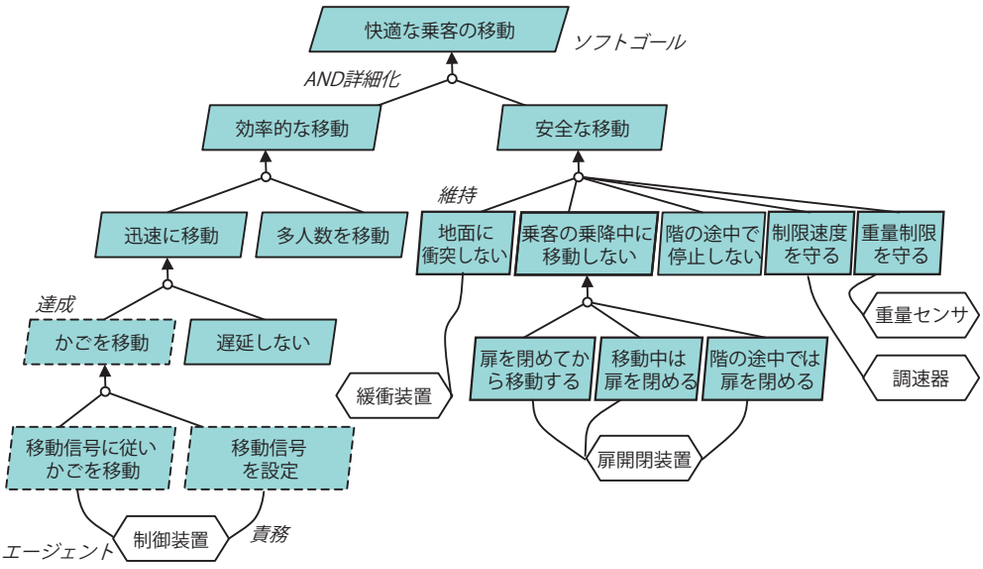

図 -2　KAOS ゴールグラフの例

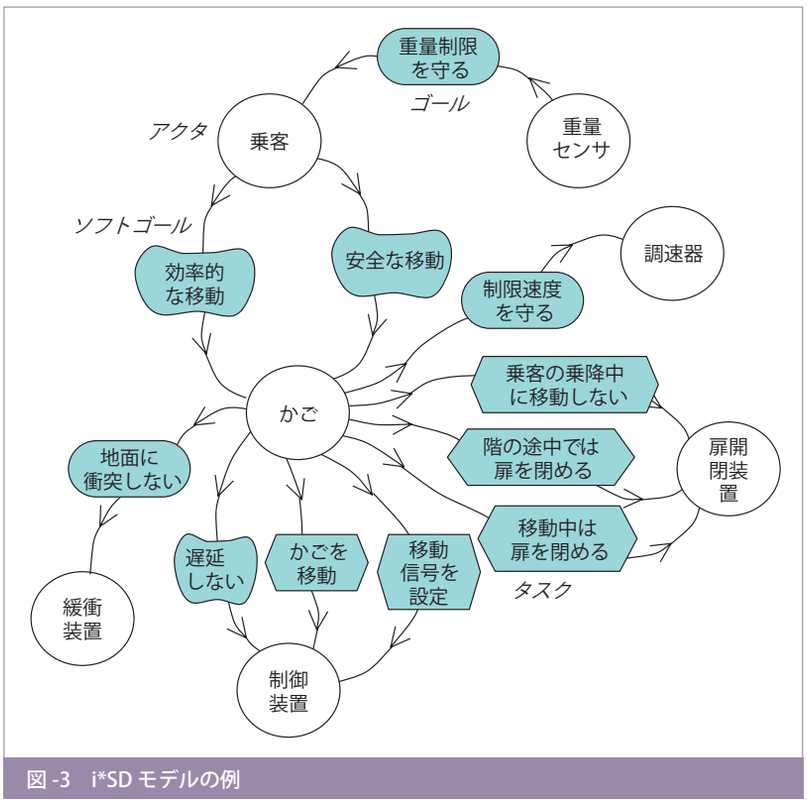

図 -3　i*SD モデルの例

は，上位ゴールとしてのゴール，タスク，資源を実現する手段としての下位ゴールであるゴールやタスクに分解する．ソフトゴールは手段目的分解では扱わない．貢献関係では，タスク，資源などの下位ゴールがどのソフトゴールに貢献するかを記述する．つまり代替案としてのタスクや資源をソフトゴールへの貢献度を基準にして評価することができる．

図 -2 のエレベータ制御に対応する SD モデルの例を**図 -3** に示す．この図では資源は出現していない．図 -2 では記述していなかった「乗客」やエレベータの「かご」をアクタとして記述している点を注意しておく．この理由はソフトウェアに対応するエージェントだけを KAOS で記述するためである．

■ NFR フレームワーク

NFR フレームワーク（Non Functional Requirements Framework）では非機能要求をソフトゴールで表現する．ソフトゴールには次の 3 種類がある．ソフトゴールとそ

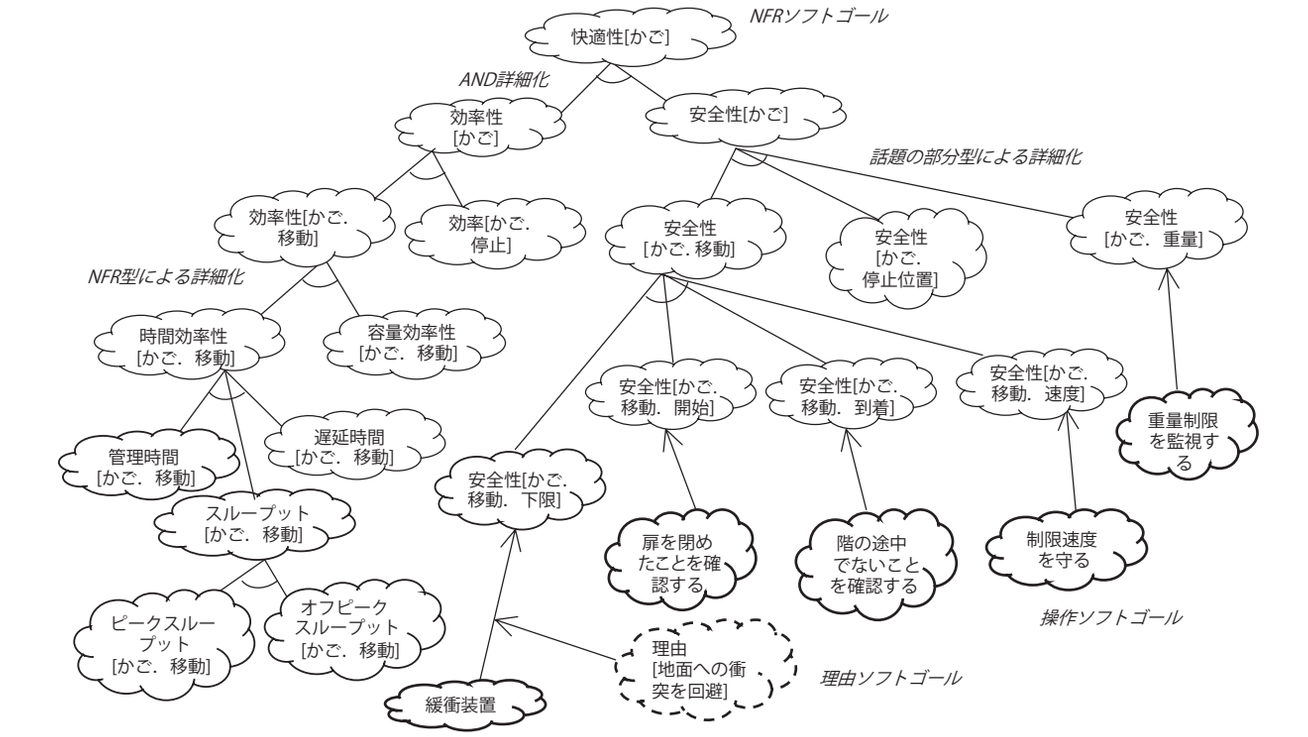

図 -4　NFR フレームワーク SIG の例

の相互依存関係はソフトゴール依存関係グラフ（SIG）を用いて表現する．

- NFR ソフトゴール：満足化（Satisfice）の対象となる非機能要求を表現する．ここで，現実世界では大域的な最適化問題では厳密な解を簡単には発見できないことに着目して，最適化が困難な状況であっても十分に良い解を見つけるための手続きを満足化という[6]．
- 操作ソフトゴール：満足化を達成する技術を表現する．操作ソフトゴールを対応付けることを操作化（Operationalization）という．操作ソフトゴールが機能ゴールに対応付けられる．
- 理由ソフトゴール：意思決定や相互依存関係の根拠を理由（Claim）ソフトゴールで表現する．

NFR フレームワークでは，まず対象システムの制約条件を NFR ソフトゴールで明らかにする．次にソフトゴールごとに下位ゴールに分解することにより，SIG を作成する．非機能要求に対する NFR ソフトゴールを操作ソフトゴールに対応付けることにより対象システムの機能を記述する．

SIG では終端節点を評価・選択して親ソフトゴールを満足化できることを確認することにより，対象システムに関する設計判断を選択された終端節点で表現する．

つまり機能要求の分析では，非機能要求分析により指定された非機能ソフトゴールを満足化できる代替案を選択する．非機能ソフトゴールや機能ゴールが対立すると，非機能要求を満足化するかどうかを判断基準にして対立関係を解消する．

またソフトゴールには，NFR 型名と話題名，優先順位を次のようにして記述する．

！安全性　［かご . 重量 ］{ 重要 }

ここで，NFR 型名は「安全性」，話題名は「かご」，「重量」はかごの属性である．また優先順位「重要」がついているソフトゴールであることを！記号で示している．

エレベータ制御に対する SIG の例を**図 -4** に示す．この例から分かるように，SIG では NFR 型と話題の構造を利用してソフトゴールを詳細化している．たとえば NFR 型である時間効率性については管理時間，スループット，遅延時間などに詳細化されている．これらの NFR ソフトゴールを評価基準として，かごの制御装置が実現する操作ソフトゴールの代替案を考案することになる．また理由ソフトゴールによって SIG 上で操作化を明示的に説明できる．このように代替案の評価やソフトゴールの詳細化が手順化されている点に SIG の特徴がある．

	KAOS	i* フレームワーク	NFR フレームワーク
ゴールグラフの構成要素	ソフトゴール 機能ゴール エージェント 領域特性	ソフトゴール ゴール タスク 資源 アクタ	NFR ソフトゴール 操作ソフトゴール 理由ソフトゴール
ゴール属性	構成要素名 構成要素種別 時相論理式 ゴールパターン （維持，回避，達成，中断）	構成要素名 構成要素種別	非機能要求名 構成要素種別 話題 優先順位 満足化の値
関係	AND/OR 関係 操作化関係 責務関係	AND/OR 関係 手段目的関係 タスク分解関係 貢献関係	AND/OR 関係 操作化関係 理由関係 貢献関係

表 -2　主なゴール指向手法の比較

■ ゴール指向手法の比較

表 -2 に KAOS，i* フレームワーク，NFR フレームワークにおけるゴールグラフの構成要素とその関係をまとめた．

KAOS の特徴は，4 種のモデルを用いることによる仕様から設計への段階的な追跡性と，時相論理を用いた仕様の形式検証ができる点である．逆にこれらの多様なモデルと時相論理の知識がなければ使いこなせないということになる．

i* フレームワークの特徴は，アクタ間の依存関係を柔軟に分析でき，ソフトウェアだけでなく組織や業務の分析にも適用できる点である．また NFR フレームワークの概念を用いることでアクタ内のゴールとその達成手段の関係を詳細化できる．これまでの筆者による大学院での教育経験からいうと i* フレームワークを習得する上での課題には次の 2 つがある．まず，アクタ間の依存関係の矢印の向きが，従来のデータフローなどの向きと逆になっているため初心者には理解しがたいことである．また SD モデルや SR モデルを作成してもどのようにして図の十分性や完全性を確認していいか分からないとの声も多い．これらについての具体的な規約作りが必要であろう．たとえば，i* フレームワークに基づいて形式仕様記述言語を作成するという手法も提案されている．一方で，ゴールモデルでは必ずしも完全性を求めるのではなくシステムのスコープなどの概略だけを記述すればよいという意見もあるかもしれない．この立場では，KAOS などの手法は厳密すぎて適用しない方がよいということでもある．

NFR フレームワークの特徴は，ゴールグラフの簡便性とソフトゴールの満足化の機構である．一方，ノードが多くなりゴールグラフが複雑になるとツールがないと効率的に管理できなくなる可能性もある．

さて図 -1，図 -2，図 -3 でも分かるように，これらのゴール指向手法のゴールグラフ間には類似性がある．たとえば i* フレームワークの SR モデルでは NFR フレームワークと同様の貢献関係によって代替案としてのタスクを比較評価できる．どの手法を用いるかは何のためになぜゴールを分析するのかというゴール指向要求定義の目標をまず定義する必要がある．その目標を達成するための手段としてどのゴール指向手法が適しているかを判断することが大切である．

■ ゴール指向の適用例

KAOS の事例研究には，地下鉄システム（San Francisco Bay Area Rapid Transit, BART）の例 [1] があり，ビジネス情報システムだけでなく，組込みシステムにもゴール指向が適用できることが分かる．

i* フレームワークはセキュリティ要求分析を始め数多くの応用が研究されているゴール指向手法である．たとえば，航空業務オペレーションへの適用では，55 個のアクタと 186 の依存関係を持つ SD モデルを作成したところ 578 個の要求を抽出でき仕様書の大部分を自動作成できた事例が報告されている [7]．

NFR フレームワークの適用例としては，セキュリティ要求分析，効率性要求分析，クレジットカードシステム，業務プロセスの効率性分析，ソフトウェアアーキテクチャ分析などがある [3]．

■ ゴール指向の課題

Rolland[5] らはゴール指向の適用経験から，以下のような課題を挙げている．

ゴールという抽象的な概念を業務専門家が扱うことは難しいため，ゴールを下位ゴールに具体化するための手法を研究する必要がある．特にゴールを実現するためには操作ゴールや代替ゴールの抽出を支援する手法が必要である．

ツール名	開発機関	概要	参考
Objectiver	Objectiver 社	KAOS 記法の図式エディタ IEEE std 830-1998 文書化支援	http://www.objectiver.com
OME	トロント大学	NFR，i* 記法の図式エディタ NFR の満足化評価	http://www.cs.toronto.edu/km/ome/
Si* Tool	トレノ大学	Secure Tropos モデルの図式エディタ Secure Tropos 仕様の解析	http://sesa.dit.unitn.it/sttool/download.php
NFR-Assistant	テキサス大学	starUML のステレオタイプを用いた NFR フレームワークの SIG の図式エディタ	http://www.utdallas.edu/~supakkul/tools/softgoal-profile/softgoal-profile.html
QFD/Capture	International Techne Group 社	品質機能展開表の表エディタ 評価支援	http://www.qfdcapture.com/default.asp
ElicitO	マンチェスター大学	FR と NFR 抽出 ISO9126 に基づく品質モデル 評価支援	Al Balushi et al., ElicitO : A Quality Ontology-Guided NFR Elicitation Tool, REFSQ 2007

表 -3　主なゴール指向関連ツール

また，システム開発ではゴールがあることを仮定しているが，実際には，企業ゴールなどの場合には，理想的な状況に基づいて定義されていて現実の状況を反映していないことが多い．このため適切なゴールを発見するための手法が必要になる．また現実の状況によってゴールを評価することで反復的に変更するようなゴール管理手法も必要になる．

さらに初期ゴールは不正確で概略的であり複数の解釈ができることが多いため，ゴール指向手法では，ゴールの形式化を支援することも必要になる．

従来技術との比較

製造業では，従来から製品の品質特性を扱う品質機能展開（QFD）や Goldratt の制約理論（TOC, Theory Of Constraints）の論理思考プロセスなどでゴール指向に類似する手法が使われている．たとえば，QC（Quality Control）手法でよく知られている特性要因図（魚の骨図）は，縦か横かという違いを除けばゴール階層と同じであることに気づく [8]．

QFD（Quality Function Deployment）や論理思考プロセスとゴール指向の関係についての詳しい比較については文献 4）を参照していただくことにして，ここでは簡単に類似点と違いを説明する．

QFD は「新製品開発における顧客要求を設計に活かし，設計の意図を生産段階に伝える」ために開発された技術であり，さまざまな品質表を用いて品質特性を製品構成要素だけでなく業務プロセスへも展開できるという特徴がある．表の構成要素をゴール，構成要素間の関係をゴール関係とすれば，品質表でゴール階層を表現できる．また品質特性ウェートにより品質を評価できる．品質特性関連表では品質特性間の関係を評価することもできる．これらの点でもゴール指向におけるゴールの優先順位付けやソフトゴール間の依存関係が品質表で記述できることが分かる．QFD をソフトウェア品質管理へ適用した先駆的な研究もある [9]．

論理思考プロセスでは，制約理論に基づく生産システムの変革案を製造現場に導入していくために 6 つの図式を持つ．現状分析ツリー，未来実現ツリー，ネガティブブランチ，前提条件ツリー，移行ツリーにより，業務活動のゴールや手段の依存関係を階層的に記述する点はゴールグラフと同じである．また対立解消図では，ゴール，ゴール達成要件，その前提との関係を分析する．ネガティブブランチでは未来実現ツリーで想定されるよくない出来事の可能性としてのリスクを抽出することができる．

ゴール指向ツールの紹介

KAOS，i* フレームワークや NFR フレームワークなどの代表的なゴール指向ツールと関連する品質要求定義ツールをまとめて**表 -3** に示す．KAOS 記法の図式エディタである Objectiver には IEEE std 830-1998 に従った要求仕様書を生成する文書化支援機能がある．トロント大学が開発した OME では i* と NFR フレームワークを融合しており，SR モデルを NFR フレームワークで記述できるだけでなく評価もできる．トレノ大学の Si*Tool は i* を拡張してセキュリティ要求などを記述できる Secure Tropos モデルを編集できる．テキサス大学の NFR-Assistant は当初 Prolog で開発されていたが starUML のステレオタイプにより SIG を記述するツールが提供されている．

また品質要求定義ツールとして，International Techne Group 社による品質機能展開を支援するツール QFD/Capture や ISO9126 の品質モデルを作成評価できるマンチェスター大学の ElicitO などがある．

Si*Tool や NFR Assistant などはダウンロードして利用できるが，日本語を表記できないという課題がある．これらのゴール指向ツールの日本語化が望まれる．

今後の展望

ソフトウェア開発における非機能要求の重要性が高まるに従ってゴール指向要求工学の必要性が認識され多様なゴール指向手法が提案されている．今後はこれらのゴール指向手法の統合だけでなく，高信頼システム開発，サービス開発，法制度など社会システム開発への応用も期待される．

特に高信頼システム開発では，アーキテクチャ設計とゴール指向手法との統合も必要になる．たとえば，アーキテクチャ設計ではシステム構成要素間の接続関係を定義する必要がある．システム構成要素をアクタだと考えれば，アクタ間の接続関係をアーキテクチャでは定義することになる．本稿で説明したように，ゴール指向手法ではアクタ間の関係をソフトゴールで定義する．またアクタが提供する機能が持つべき品質特性を非機能要求として定義することができる．このようにアーキテクチャ設計はシステム構成要素間の関係を定義しなければ具体化できないし，ソフトウェア要求はソフトウェアの外部境界としてのシステム構成要素が明確になっていなければ具体化できない．また自動車などに見られるようにデバイス自身に組み込まれるソフトウェアが増加している．したがってゴールとアーキテクチャの協調デザイン手法が必要になる．今後もこれらの手法に関する一層の研究開発を期待したい．

参考文献

1）van Lamsweerde, A. : Goal-Oriented Requirements Engineering : A Guided Tour, Proceedings RE'01, 5th IEEE International Symposium on Requirements Engineering, pp.249-263 (2001).
2）i* webpage, http://www.cs.toronto.edu/km/istar/
3）Chung, L., Nixon, B., Yu, E. and Mylopoulos, J. : Non- Functional Requirements in Software Engineering, Academic Publishers (1999).
4）山本修一郎：〜ゴール指向による〜システム要求管理技法，ソフト・リサーチ・センター (2007).
5）Rolland, C. and Salinesi, C. : Modeling Goals and Reasoning with Them, in Aurum, A. and Wohlin, C. eds. "Engineering and Managing Software Requirements", Springer (2005).
6）Simon, H. A. : The Sciences of the Artificial, MIT Press (1981). 稲葉元吉，吉原英樹（訳）：システムの科学，パーソナルメディア社 (1987).
7）Ncube, C., Lockerbie, J. and Maiden, N. : Automatically Generating Requirements from i*. Models : Experiences with a Complex Airport Operations. System, REFSQ 2007, LNCS 4542, pp.33-47 (2007).
8）山本修一郎：特性要因図とゴール指向，ビジネスコミュニケーション，http://www.bcm.co.jp/site/youkyu/youkyu30.html
9）大森　晃：ソフトウェア品質管理への品質展開アプローチ―概念的枠組みと方法論―，情報処理学会論文誌，Vol.31, No.10, pp.1474-1485 (Oct. 1990).

（平成 20 年 1 月 24 日受付）

山本修一郎（正会員）　yamamotosui@nttdata.co.jp

1979 年名古屋大学大学院工学研究科情報工学専攻修了．同年日本電信電話公社入社．2002 年（株）NTT データ技術開発本部副本部長．2007 年同社，システム科学研究所所長．ソフトウェア工学，ユビキタスコンピューティング，知識創造デザインの研究に従事．情報処理学会業績賞，通信協会前島賞など受賞．博士（工学）．