JAIST Repository
https://dspace.jaist.ac.jp/

| | |
|---|---|
| Title | 法令実働化情報システムのアカウンタビリティ |
| Author(s) | 落水，浩一郎 |
| Citation | |
| Issue Date | 2007-09-07 |
| Type | Presentation |
| Text version | publisher |
| URL | http://hdl.handle.net/10119/8266 |
| Rights | |
| Description | 北陸先端科学技術大学院大学 21世紀COEシンポジウム「検証進化可能電子社会」 = JAIST 21st Century COE Symposium “Verifiable and Evolvable e-Society”，開催：2007年9月6日～7日，開催場所：キャンパス・イノベーションセンター東京 国際会議室(1F)，2007年9月7日（金），「JAIST-COE シンポジウム：法令工学の可能性と展望」発表資料 |

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

# 法令実働化情報システムの アカウンタビリティ

## ソフトウェアアカウンタビリティの定義と実現

落水浩一郎
北陸先端科学技術大学院大学
情報科学研究科

法令工学の展望と技術 シンポジウム
2007年9月7日 東京田町キャンパスイノベーションセンター

# 内容

- 背景と目標
- ソフトウェアアカウンタビリティ定義の立場と関連研究
  - ゴール指向要求分析
  - 法理論
- 基本概念の形式化
  - ソフトウェアアカウンタビリティ木
  - ソフトウェアアカウンタビリティモジュール
- ソフトウェアアーキテクチャ
  - ソフトウェアアカウンタビリティモジュールを既存の情報システムに装着する手段
- 事例研究
  - 実現可能性の確認

# 我々は**Trustworthy e-Society**に生きる必要がある

- われわれの日常生活は, 電子社会システムに強く依存している
- 電子社会システムはdependable でtrustworthyでなければならない
- 21世紀 COEプログラム(片山) 「検証進化可能電子社会」
  - 5つの安心性要件
    - 正当性
    - アカウンタビリティ
    - セキュリティ
    - 耐故障性
    - 進化性
- われわれの研究目的
  - ソフトウェアアカウンタビリティ
  - 進化容易性

# 社会規則と 法実働化情報システム(LEIS)

- 社会規則 (法理論では規範（Norm ）という)
  - 法律，条例，規則などの総称
- Law-Enforcing Information System (LEIS)
  - ある特定の社会規則の適用を支援する
  - 社会規則を完全に満たすように構築されている
  - 社会規則に従って電子社会システムが正しく構築されていることを保証でき, かつ確認できる必要がある
  - 社会規則の改定に応じて, 迅速に, 低いコストで進化させうる

# ソフトウェアアカウンタビリティ
（タイプ３　ソフトウェアアカウンタビリティ）

- LEISがシステム利用者に，LEIS が何故そのような決定や計算をしたかを説明できる
- 言葉をかえれば, LEIS は, LEIS によってなされた決定や計算結果に対して利害関係者が疑問を持ったとき, その質問に答えることができる
- LEIS は, 質問に対する答えを, 実行履歴と社会規則をもとに作成できることが望ましい
  （何をどのように適用したか)

# LEISと利害関係者の実例

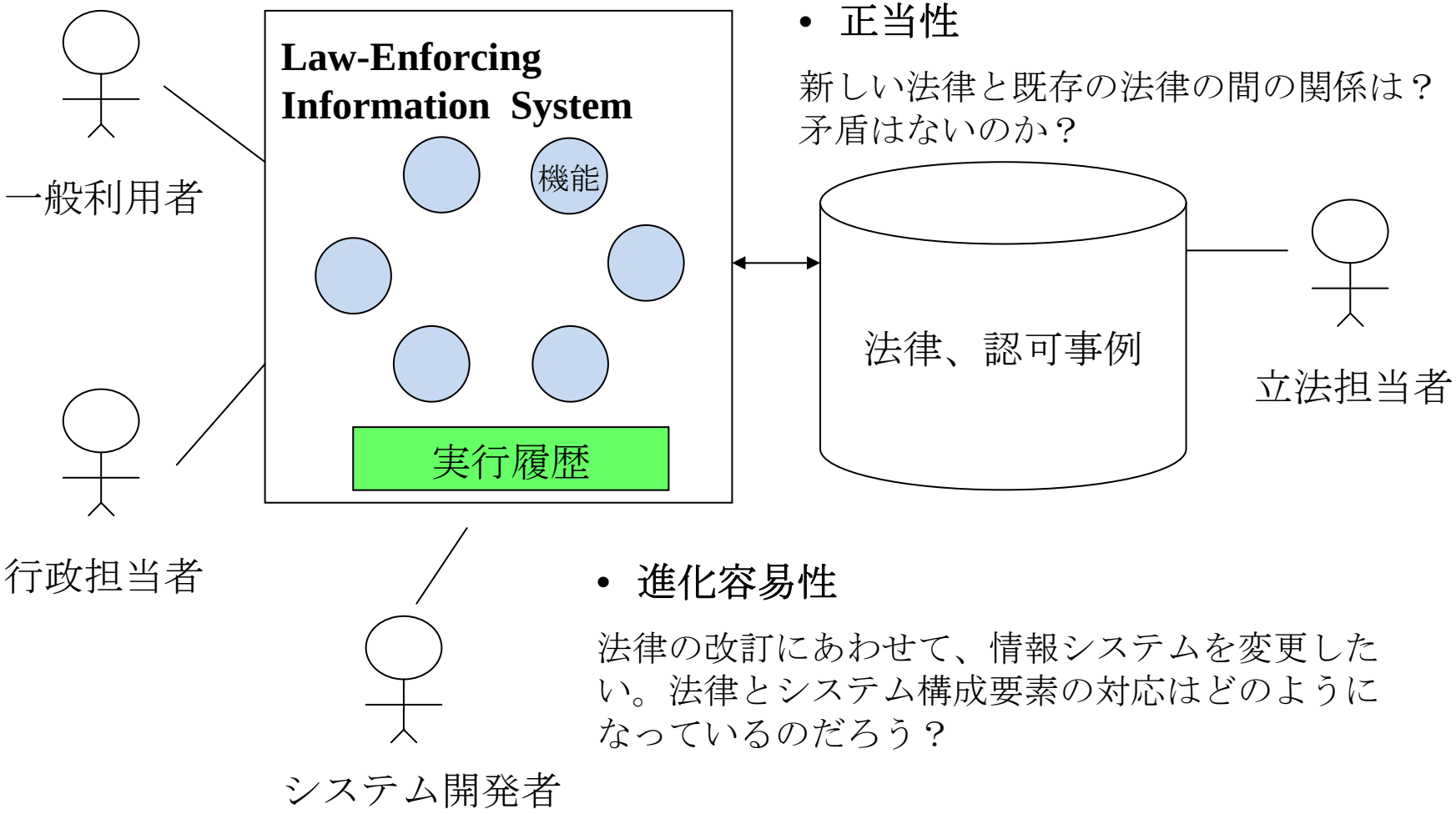
- ある大学の履修管理システム
  - 学生, 教員, 教務学生係, システム開発者
- 地方自治体システム
  - 条例制定者, システム開発者, 窓口サービス担当, 市民
- ある企業の情報システム
  - 管理者, 従業員, システム開発者

利害関係者間のシステムを通じた
コミュニケーション

# 利害関係者の典型的な型と質問の例

- アカウンタビリティ（自己説明性）

情報システムを利用して電子申請や登録を行った。システムが提示した処理結果について疑問がある。この結果はどのような法律や条令をどのように利用して許可・不許可されたのだろうか？




- 正当性

新しい法律と既存の法律の間の関係は？矛盾はないのか？

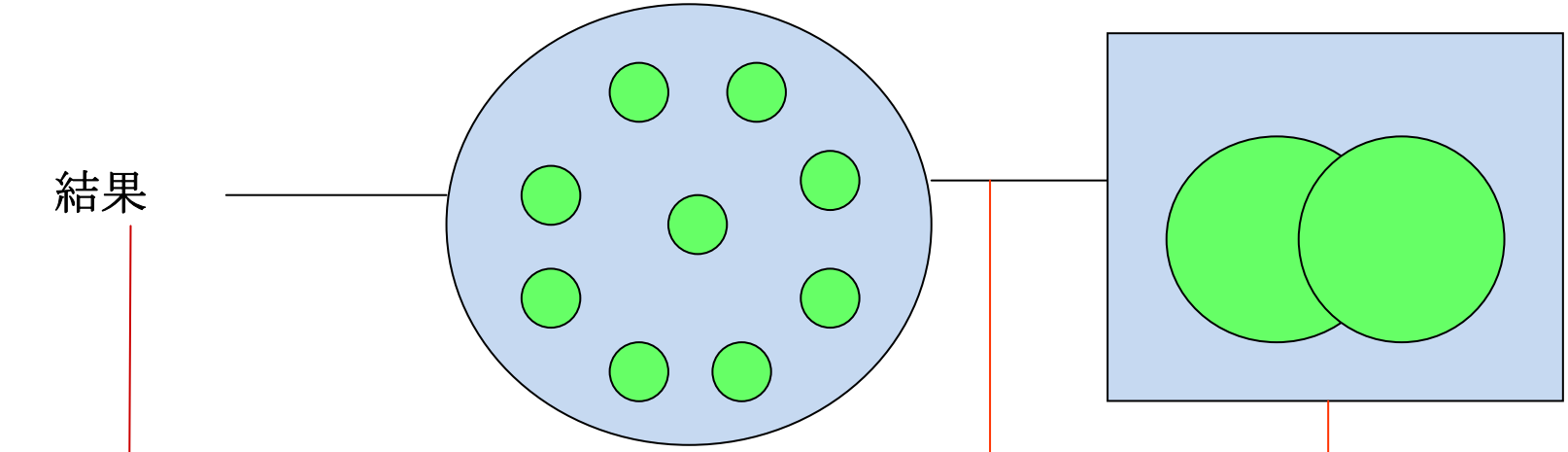
- 進化容易性

法律の改訂にあわせて、情報システムを変更したい。法律とシステム構成要素の対応はどのようになっているのだろう？

# 3種のLEISに対する利害関係者の関心

- タイプ　1
  - 社会規則そのものに関する質問
- タイプ　2
  - 社会規則とシステム構成要素の対応関係に関する質問
- タイプ　3
  - システムの実行結果に関する質問

# LEISに対する3種類の関心




タイプ３

理由説明：かくかくしかじかの理由でこう決定した（実行履歴→理由説明）

タイプ２

対応の説明：この法律はここに、このように実現されている（対応構造→妥当性説明）

タイプ１

法律自体の説明：どこに矛盾があるかを説明できる

# 内容

- 背景と目標
- ソフトウェアアカウンタビリティ定義の立場と関連研究
  - ゴール指向要求分析
  - 法理論
- 基本概念の形式化
  - ソフトウェアアカウンタビリティ木
  - ソフトウェアアカウンタビリティモジュール
- ソフトウェアアーキテクチャ
  - ソフトウェアアカウンタビリティモジュールを既存の情報システムに装着する
- 事例研究
  - 実現可能性の確認

# ソフトウェアアカウンタビリティの定義にあたって

- ソフトウェア工学的立場からの考察
  - ゴール指向要求分析
  - ゴール指向木
- 法理論的立場からの考察
  - 法システム：規範と行為（Norm and Act）
  - 規範間の静的/ 動的関係
- (ソフトウェア) アカウンタビリティ木
  - ゴール指向木に基づいて構成し, 法理論によって型付けする

# 要求工学分野における関連研究

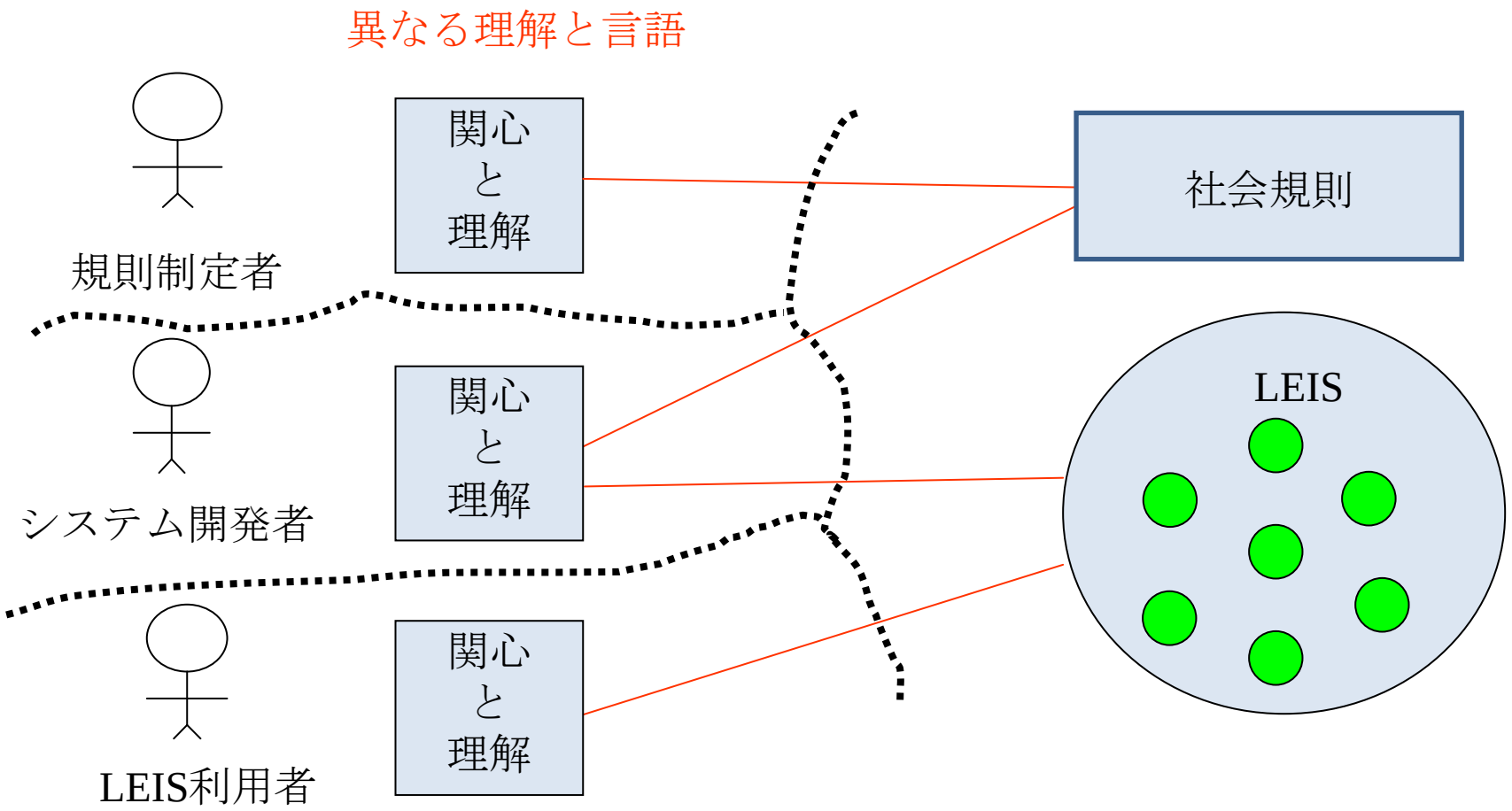
- 要求獲得活動は利害関係者の学習プロセスである
  - すべての利害関係者は, 対象領域やLEISに対して, 直接または間接に関心を持つ
  - 利害関係者は, 学習の過程を経て, 対象世界の理解に到達する
  - 理解した結果を彼等自身の言語で表現する

# ソフトウェアアカウンタビリティ定義にあたってのわれわれの立場

- 各利害関係者は
- 社会規則が制定され, システムが開発される前に,
  - 社会規則かつ/またはシステムに彼等自身の関心を持つ
  - 関係する世界を学習し理解しようとする
  - 学習の結果を彼等自身の言語で表現する
- 社会規則が制定され, システムが開発された後に,
  - 学習結果にふたたび関心を持つ

# 利害関係者は独自のセマンティクスと言語をもつ

- 種々の利害関係者はシステム開発の前／後に、システムに関する独自の関心を、彼等自身の言語で表現する

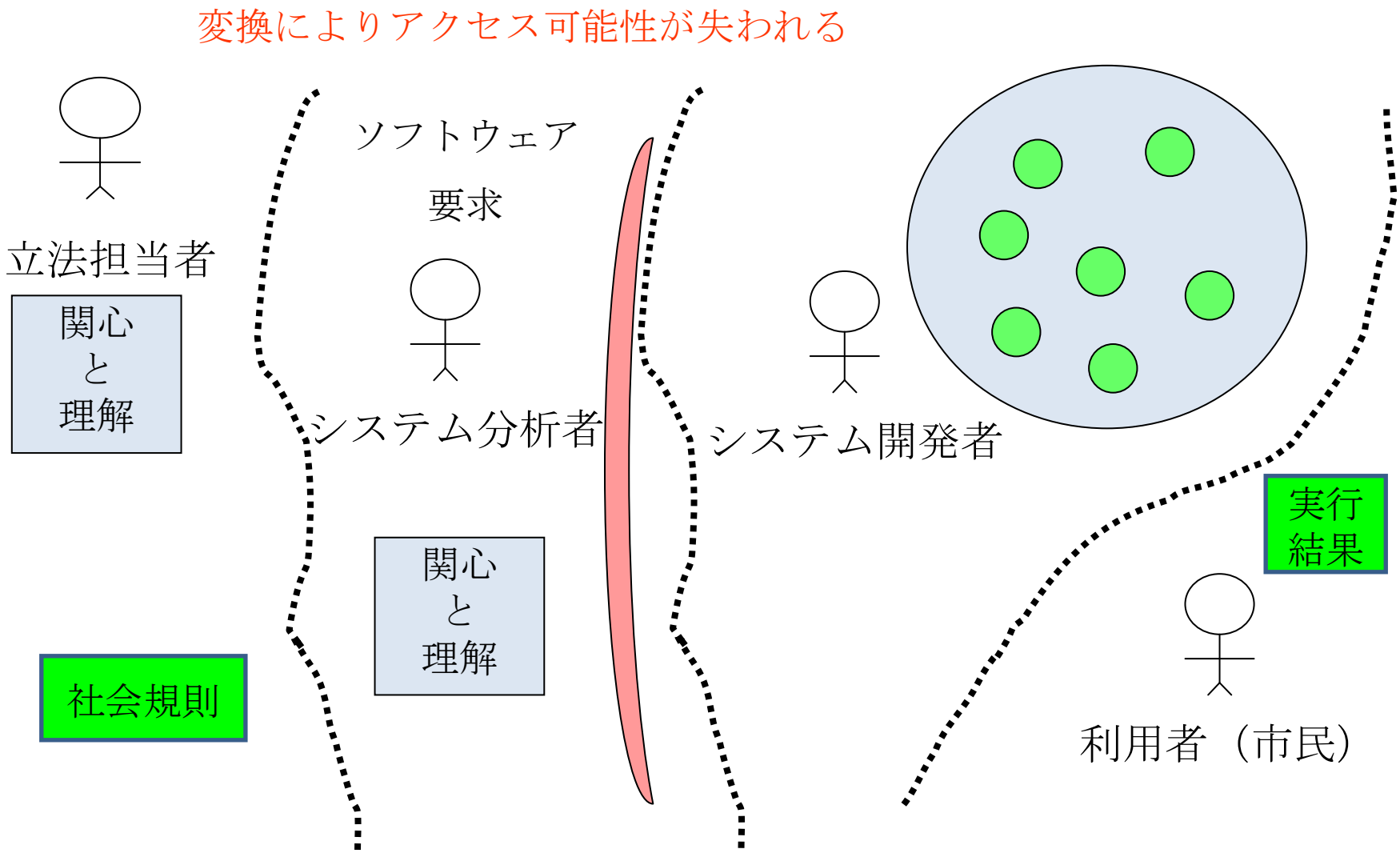


# ソフトウェアアカウンタビリティの主要な情報源

- ソフトウェアアカウンタビリティ実現のためには
- これらの結果はシステム中に記録され, 必要に応じて検索されるべきである

- 伝統的な要求管理技術に従うと, 重要な情報が失われる危険性がある

# （ソフトウェア工学における）従来のアプローチ



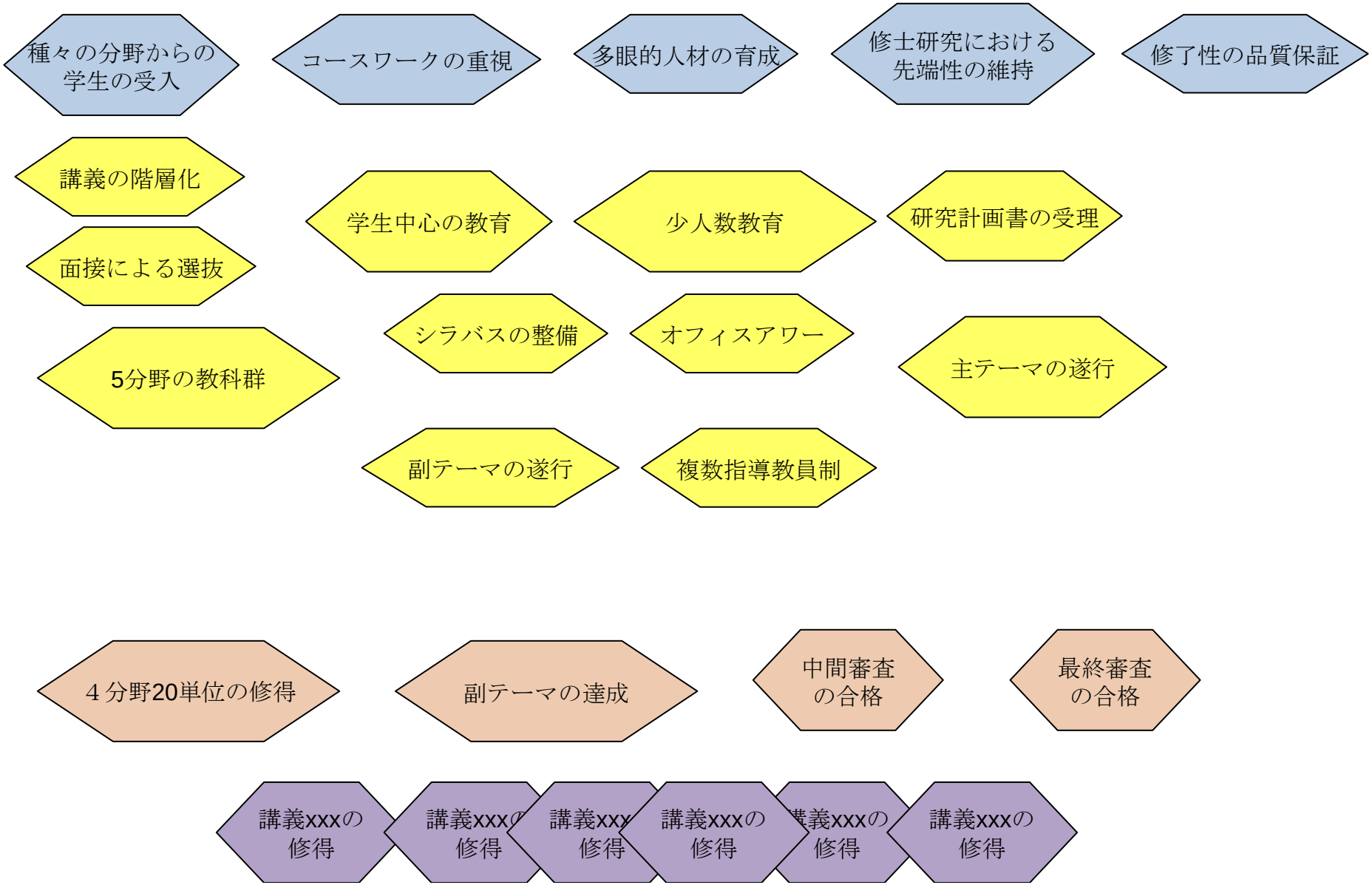
# 情報構造を編成し記録するために ゴール指向要求分析法を採用

- GORE
  - 以下の例にあるような非機能要求をゴールとしての識別することから出発して
    - ゴール：保守容易性,　ユーザビリティの良さ
  - つぎつぎにサブゴールを定義しつつAND-OR木として展開していく
  - ソフトゴール
- GORE を採用することにより, 利害関係者が理解したセマンティクスとその間の関係を表現することが可能になる

# 複数の利害関係者が理解した世界に基づくゴール木の構成

| |
|---|
| 規則を作る人が意図し、理解し、表現した世界 |
| 社会規則群 |
| Law-Enforcing Information Systemに対する機能要求群 |

# 教育システム設計者のセマンティクスの表現



# ゴールとサブゴールは以下の点で有用である

- 社会規則の進化を制御するため

# ゴール指向木は有用ではあるが<br>それだけでは十分でない

- 社会規則はたがいに関係しあっているので, 社会規則自体を構造化する必要がある
- 社会規則を分類しておくほうが良い
- そこで, 法理論の成果も適用する

# 法理論の立場からの考察

- Torstein Eckoffによる法理論
  - 法システム：規範と行為（Norms and Acts)
  - 規範: 準則, 原理, 規定, 標準, 範型, 指針, 基準などの総称
    - 指令：命令, 催告, 懇願, 助言, 警告, 約束などの総称
    - 性質決定：どのような現象が一定のカテゴリにいれられるのかを示す
    - 授権：指令・資格もしくは新しい権限を付与する権能を人に与える
  - 義務規範: 指令または指令の否定を言語的構成要素として含む規範の総称
    - 命令, 禁止, 許可, 免除の4つの下位グループを持つ

# 規範を構造化する

- 関係
  - 静的関係
    - カップリング連関
    - 意味の集積
    - 論理的関係
  - 動的関係
    - 因果連関
    - 規範的連関
    - 操作的連関

# 内容

- 背景と目標
- ソフトウェアアカウンタビリティ定義の立場と関連研究
  - ゴール指向要求分析
  - 法理論
- 基本概念の形式化
  - ソフトウェアアカウンタビリティ木
  - ソフトウェアアカウンタビリティモジュール
- ソフトウェアアーキテクチャ
  - ソフトウェアアカウンタビリティモジュールを既存の情報システムに装着する
- 事例研究
  - 実現可能性の確認

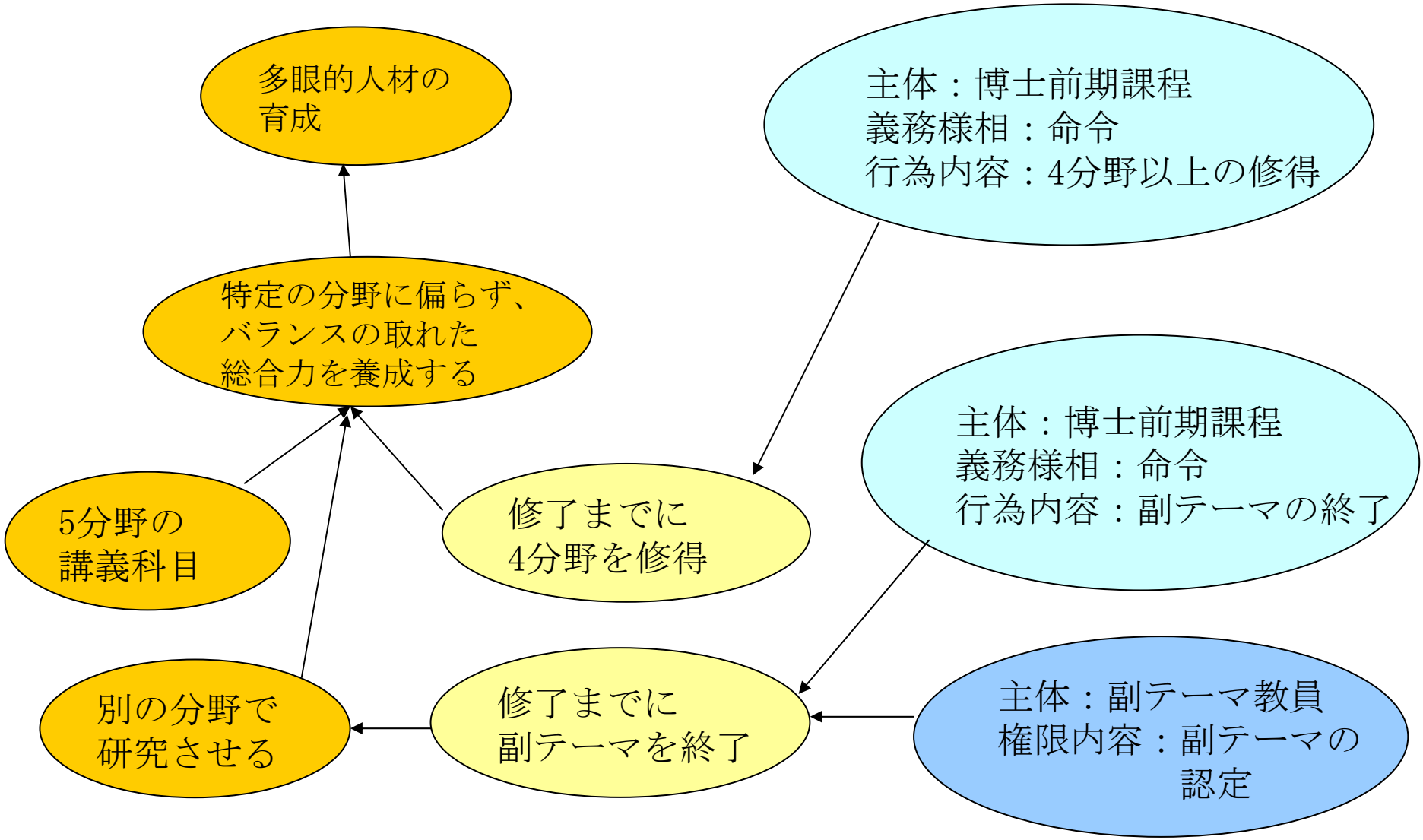
# ソフトウェアアカウンタビリティとは

- 利害関係者の学習結果と成果物をシステムが共有し、必要に応じてとりだせる

# アカウンタビリティ木

- 利害関係者によって理解されたセマンティクスをゴール指向木のノードで表現する。木の葉には社会規則の条文を置く
- 法理論に基づき,条文を型付けする．また、木の葉（条文）間には、クロスリンクとして関係を設定する
- これをアカウンタビリティ木と呼ぶ
- アカウンタビリティ木は
  - 規則制定の理由
  - 規則そのもの
  - 規則間の関係

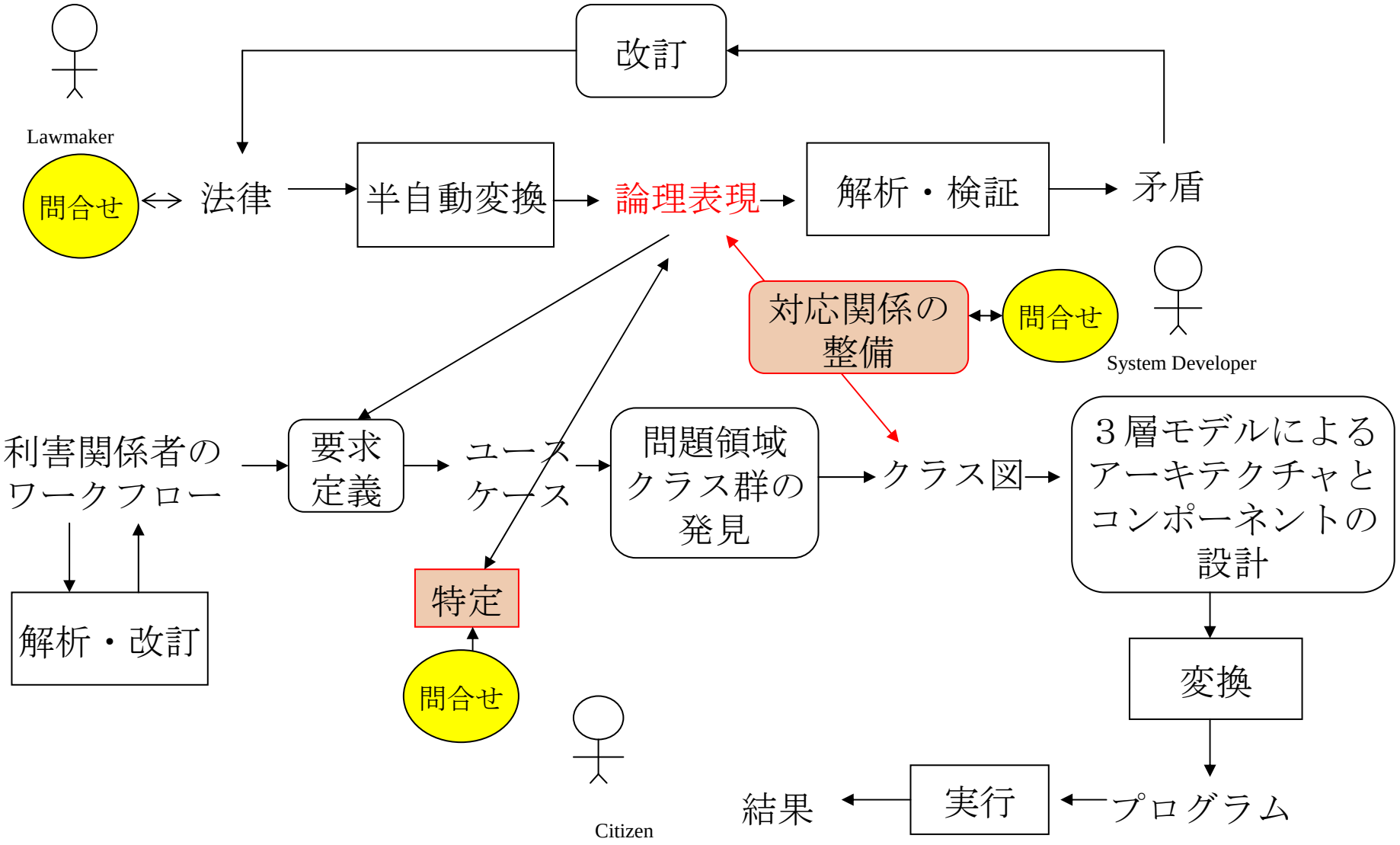
  を保持している


多眼的人材の
育成
主体：博士前期課程
義務様相：命令
行為内容：4分野以上の修得
特定の分野に偏らず、
バランスの取れた
総合力を養成する
主体：博士前期課程
義務様相：命令
行為内容：副テーマの終了
5分野の
講義科目
修了までに
4分野を修得
別の分野で
研究させる
修了までに
副テーマを終了
主体：副テーマ教員
権限内容：副テーマの
認定

# タイプ１, タイプ２ , タイプ３に対する情報源はどこに存在するのか

- **法令エンジニアリング**(LEIS開発) プロセス
  - 平文で表現された社会規則は, 自然言語処理により論理表現に変換される（島津）
  - **法令デバッギング**　論理表現は法推論機構により自動解析され, 矛盾が検出される．矛盾解消は人手により行う（東条）.
  - 法令デバッギングにおける中間表現（論理表現）はLEIS 設計の入力となる
  - MDAを適用してシステムを生成する

# LEIS開発プロセス中に存在する



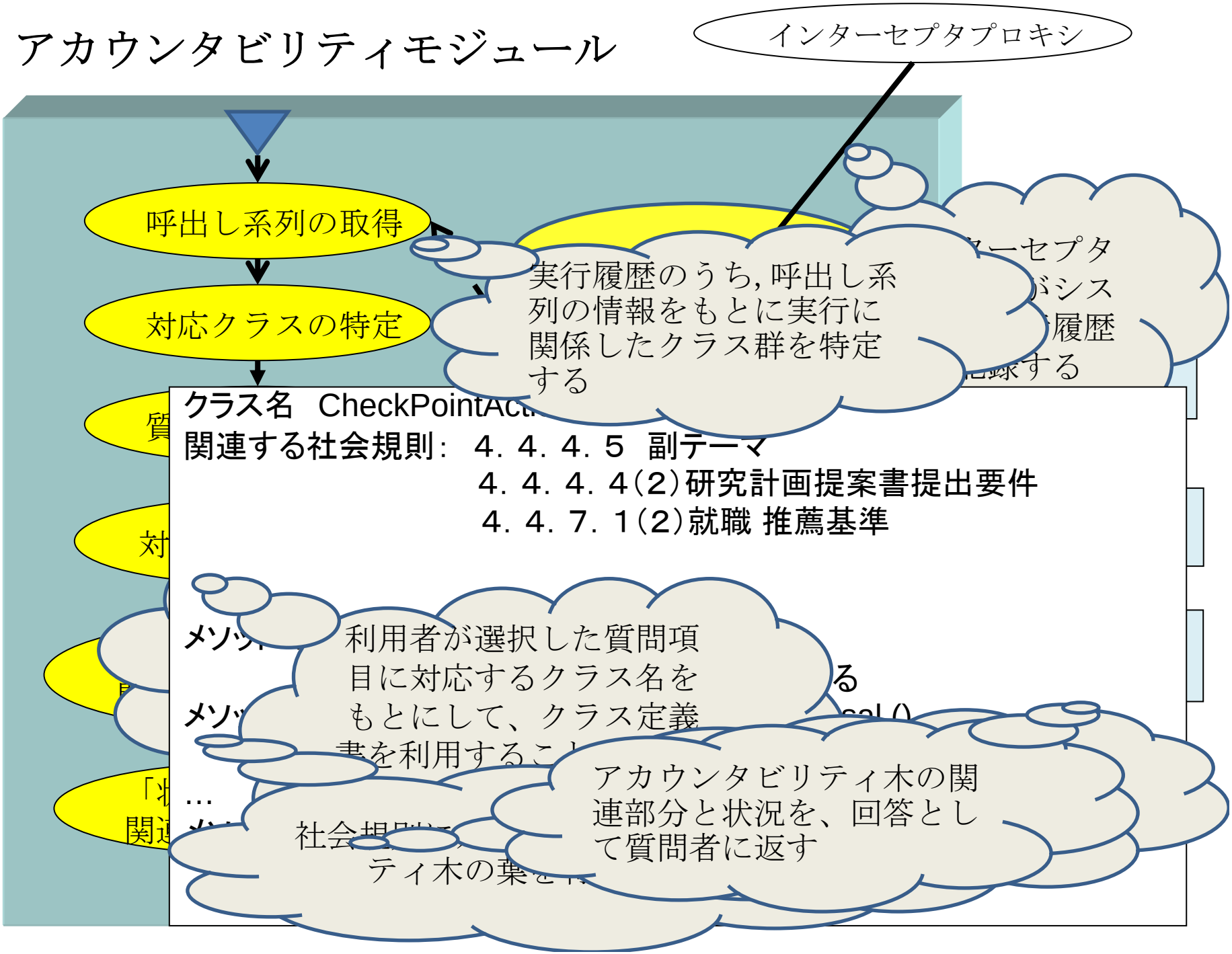
# ソフトウェアアカウンタビリティの情報源

- **論理表現**　タイプ1ソフトウェアアカウンタビリティに対する第一義の情報源
- **論理表現とクラス図の対応**　タイプ2ソフトウェアアカウンタビリティに対する第一義の情報源
- **論理表現, 論理表現とクラス図の対応, システムの実行履歴**　タイプ3ソフトウェアアカウンタビリティに対する第一義の情報源.

# アカウンタビリティ　モジュール

1. アカウンタビリティ木とアカウンタビリティ機能を持つソフトウェアモジュール
2. アカウンタビリティ機能
   - 関連する社会規則の条文を特定する機能
   - 特定された条文がどのように適用されたのかを説明する機能

# アカウンタビリティモジュール



# 回答の例

- 特定された規則と利用者の「状況」を表示

特定された規則
副テーマ提出要件

・基幹・専門・専門講義科目2科目以上の修得
・副テーマ配属後、概ね1ヶ月で修了すること

研究計画書提出要件
・副テーマが修了していること
・基幹・専門及び専門講義科目から4分野6科目以上を修得

・・・・

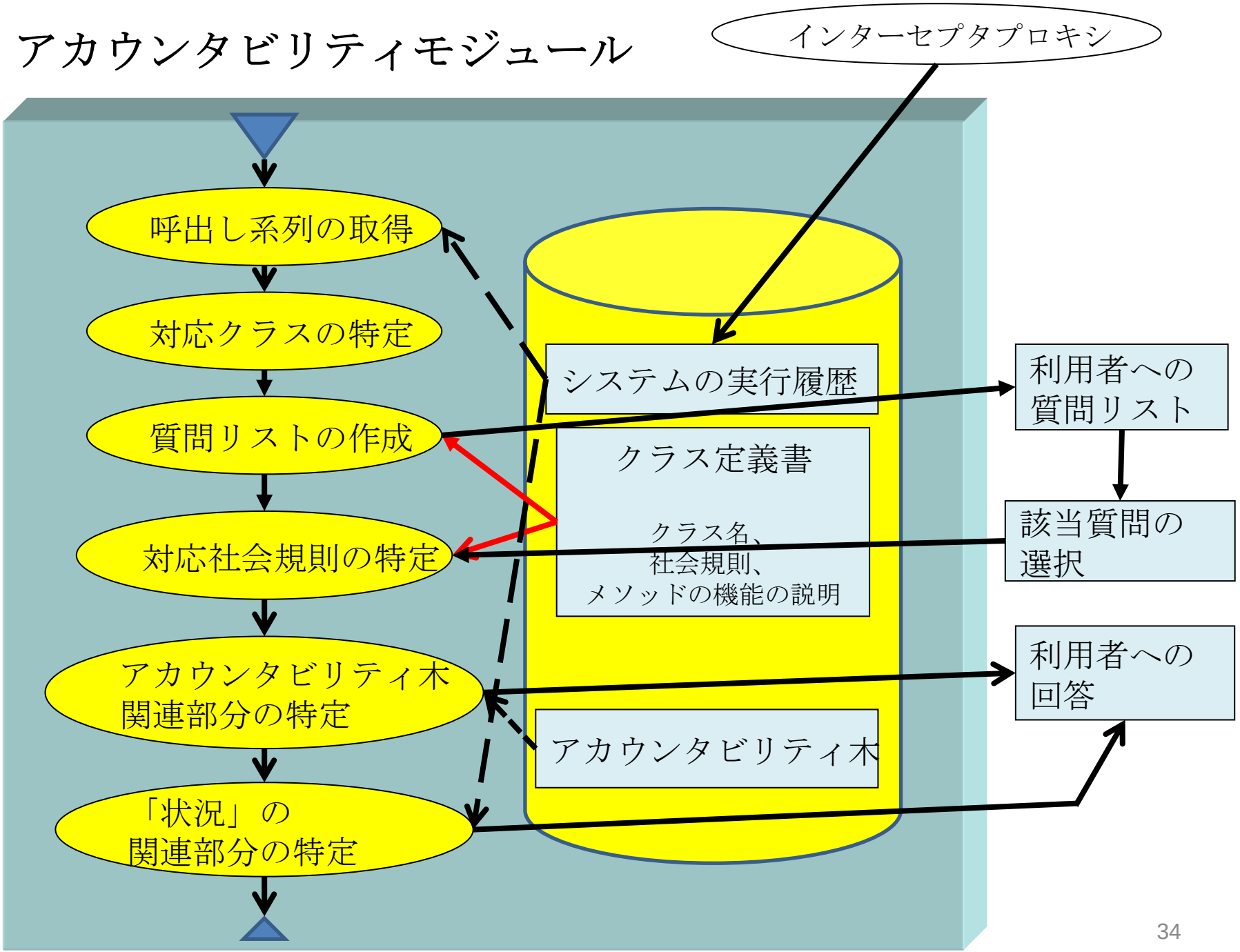
利用者の状況

学籍番号　610002
氏名　秋山　良

ア　基幹　オートマトンと形式言語　79点
イ　基幹　情報解析学特論 68点
ウ　基幹　人工知能特論　82点
エ　基幹　コンピュータネットワーク特論 65点
オ　基幹　ソフトウェア設計論　100点
オ　専門　ソフトウェア設計演習 100点

副テーマ　90点

# アカウンタビリティモジュール



# アカウンタビリティ機能の詳細

1. インターセプタプロキシがシステムの実行履歴を記録する
2. 実行履歴のうち,呼出し系列の情報をもとに実行に関係したクラス群を特定する
3. 実行に関与したクラスのメソッドから、クラス定義書を利用してメソッドの機能の説明を取り出し、それらを質問リストとして利用者に提示する。
4. 利用者が選択した質問項目に対応するクラス名をもとにして、クラス定義書を利用することにより、対応する社会規則の条文を特定する。
5. 社会規則に対応するアカウンタビリティ木の葉を特定する
6. 実行履歴のうち,引数の値は,質問者の状況を表すデータとして説明に利用する
7. アカウンタビリティ木の関連部分と状況を、回答として質問者に返す
8. 質問者は条文を状況にあてはめて結果の妥当性を確認する

# クラス定義書の例

クラス名　CheckPointAction
関連する社会規則：　４．４．４．５　副テーマ
　　４．４．４．４（２）研究計画提案書提出要件
　　４．４．７．１（２）就職 推薦基準

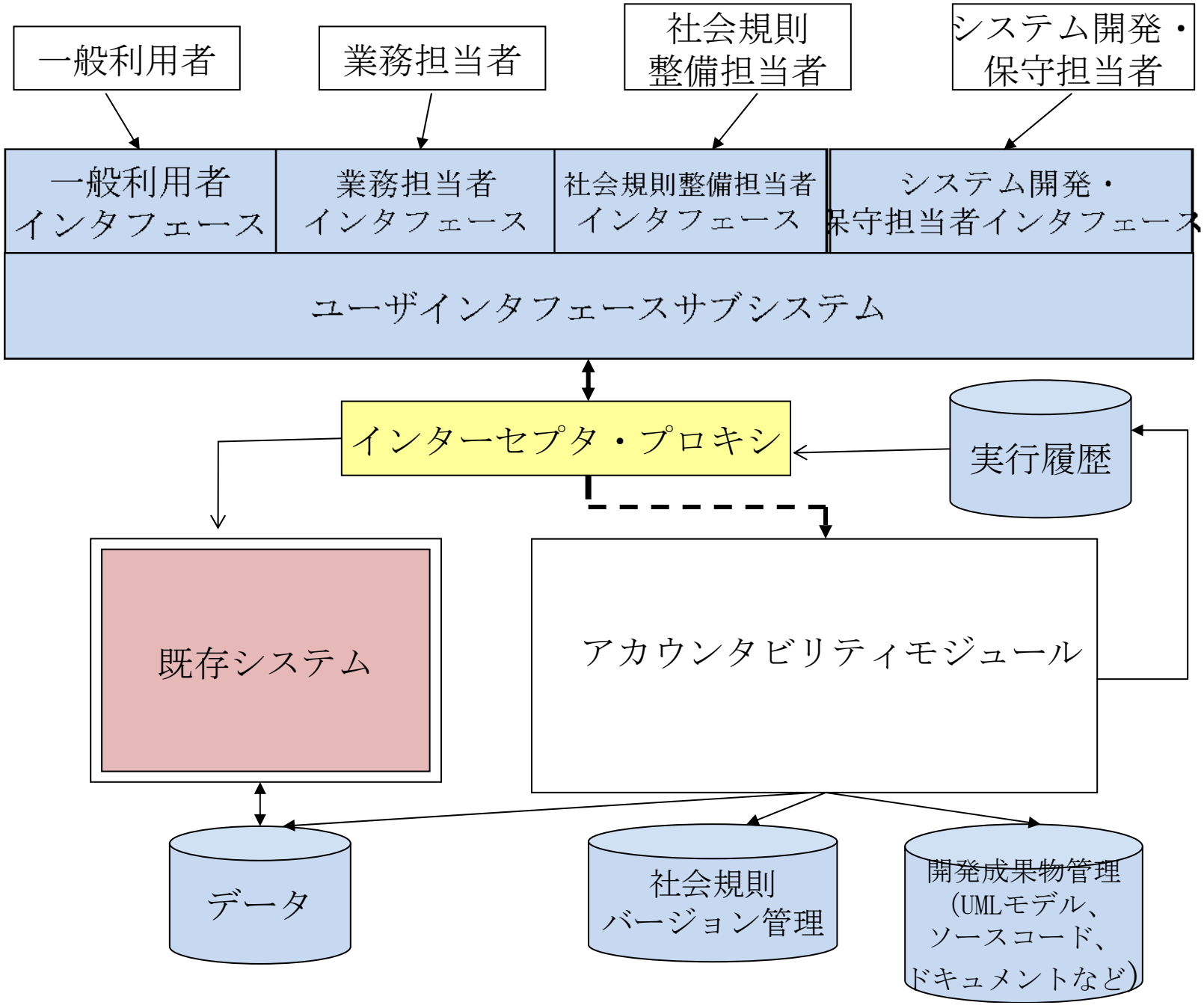
メソッドの機能の説明１：　checkSubTheme()
　　副テーマ提出要件をチェックする
メソッドの機能の説明２：　checkResearchProposal ()
　　研究計画提案書提出要件をチェックする
…
メソッドの機能の説明３：　checkJobRecommendation()
　　就職推薦基準をチェックする

# 内容

- 背景と目標
- ソフトウェアアカウンタビリティ定義の立場と関連研究
  - ゴール指向要求分析
  - 法理論
- 基本概念の形式化
  - ソフトウェアアカウンタビリティ木
  - ソフトウェアアカウンタビリティモジュール
- ソフトウェアアーキテクチャ
  - ソフトウェアアカウンタビリティモジュールを既存の情報システムに装着する
- 事例研究
  - 実現可能性の確認

# アカウンタビリティモジュールを既存の情報システムに低コストで装着するための参照アーキテクチャ

- 参照アーキテクチャの要件
  - アカウンタビリティモジュールを既存の情報システムに低コストで結合できる
  - 既存システムは最小の変更で再利用されうる
  - 大部分のLEISは, 3層モデルに基づくウェブベースシステムであることを考慮する
  - アカウンタビリティモジュールを3層モデルに結合する拡張機構が必要である
- われわれの解
  - インターセプタプロキシをユーザインタフェース層とプロセス管理層の間に置く


一般利用者
業務担当者
社会規則
整備担当者
システム開発・
保守担当者
一般利用者
インタフェース
業務担当者
インタフェース
社会規則整備担当者
インタフェース
システム開発・
保守担当者インタフェース
ユーザインタフェースサブシステム
インターセプタ・プロキシ
実行履歴
既存システム
アカウンタビリティモジュール
データ
社会規則
バージョン管理
開発成果物管理
(UMLモデル、
ソースコード、
ドキュメントなど)

# 内容

- 背景と目標
- ソフトウェアアカウンタビリティ定義の立場と関連研究
  - ゴール指向要求分析
  - 法理論
- 基本概念の形式化
  - ソフトウェアアカウンタビリティ木
  - ソフトウェアアカウンタビリティモジュール
- ソフトウェアアーキテクチャ
  - ソフトウェアアカウンタビリティモジュールを既存の情報システムに装着する
- 事例研究
  - 実現可能性の確認

# 事例研究：
# 履修管理システムにおける
# アカウンタビリティ機能の実現

- アカウンタビリティ機能を持たない履修管理システムを開発した
- アカウンタビリティモジュールを開発し, システムに結合した
- ここまでの考察結果を踏まえて実現した, アカウンタビリティ機能の能力を確認した

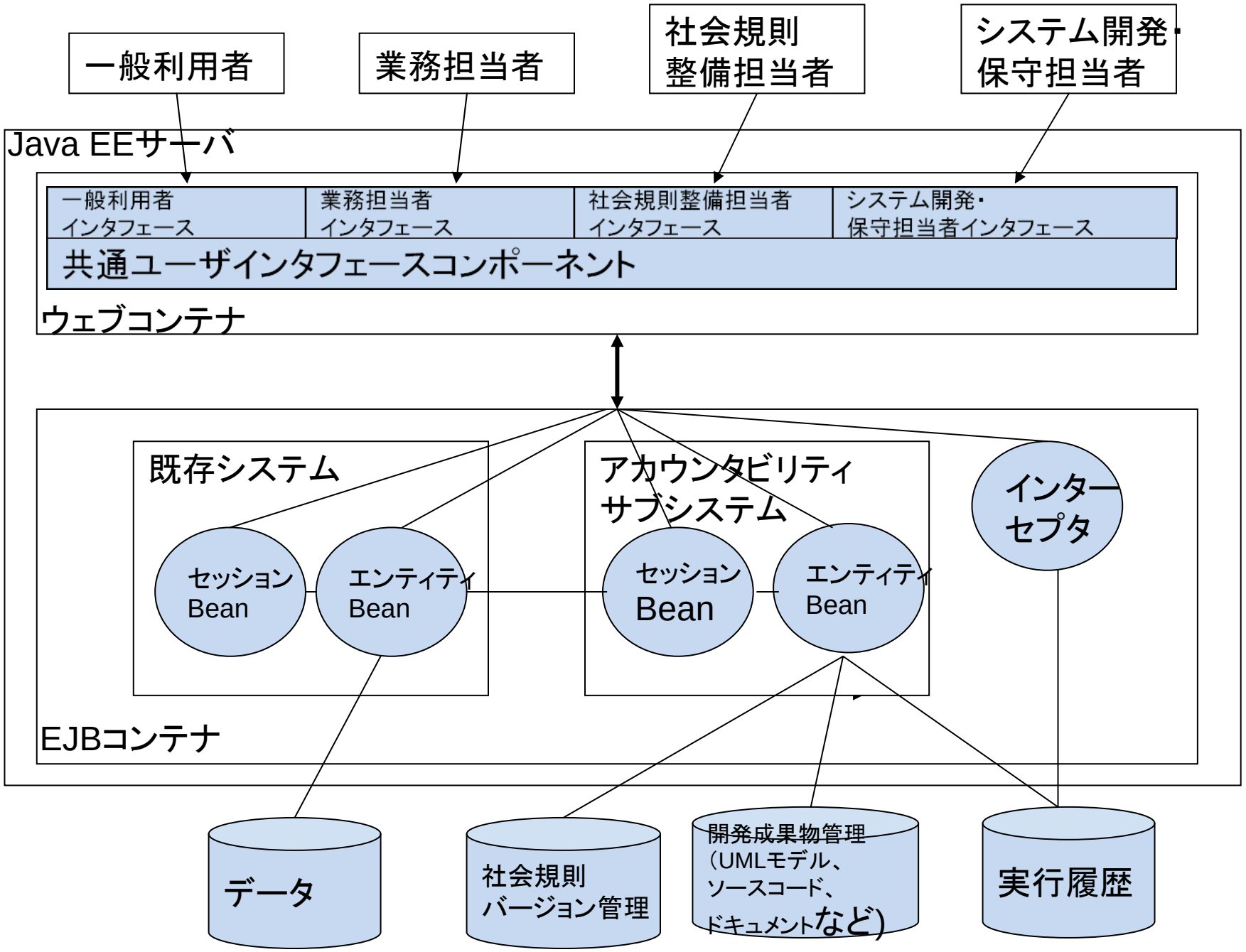
# 履修管理システムの開発

- ユースケース駆動オブジェクト指向開発方法論の利用: COMET
- ウェブベースシステム
- Java EE プラットフォーム
- JBoss Seam, ウェブアプリケーションフレームワーク
  - JSF と EJB3.0 が利用可能

- Code size
  - Java files: 約 3000 行
  - xhtml files: 約 1500 行
  - XML configuration files: 約 7000 行

# 参照アーキテクチャのJava EE への写像

- Java EEの採用
  - Java EEはその仕様中にインターセプタプロキシを含んでいる
  - アカウンタビリティモジュールの呼出しは, プロセス管理層におけるアカウンタビリティモジュールのセッションビーンズのメソッド呼出しに写像される


一般利用者
業務担当者
社会規則
整備担当者
システム開発・
保守担当者
Java EEサーバ
一般利用者
インタフェース
業務担当者
インタフェース
社会規則整備担当者
インタフェース
システム開発・
保守担当者インタフェース
共通ユーザインタフェースコンポーネント
ウェブコンテナ
既存システム
セッション
Bean
エンティティ
Bean
アカウンタビリティ
サブシステム
セッション
Bean
エンティティ
Bean
インター
セプタ
EJBコンテナ
データ
社会規則
バージョン管理
開発成果物管理
(UMLモデル、
ソースコード、
ドキュメントなど)
実行履歴

1. 呼び出し系列（メソッド名とその実行順序）の取得
    - JBoss seam　メソッド名とその実行順序
    - JDI　変数の値（実行時オーバヘッドが問題）
2. 対応クラス（メソッドが属するクラス）の特定
    - クラス定義書を利用
3. 質問リストの作成
    - クラス定義書を利用
4. クラス定義書中の条文とアカウンタビリティ木の葉に存在する条文の対応

– 条文の特定は可能：　ベクトル空間法による特定の予備実験
– どのように適用されたかの説明：目標とはするが、当面は対応する条文と質問者の状況を併記して表示し、質問者に確認させるアプローチをとる
    - 固定質問方式により効果確認済み

5. 状況の説明

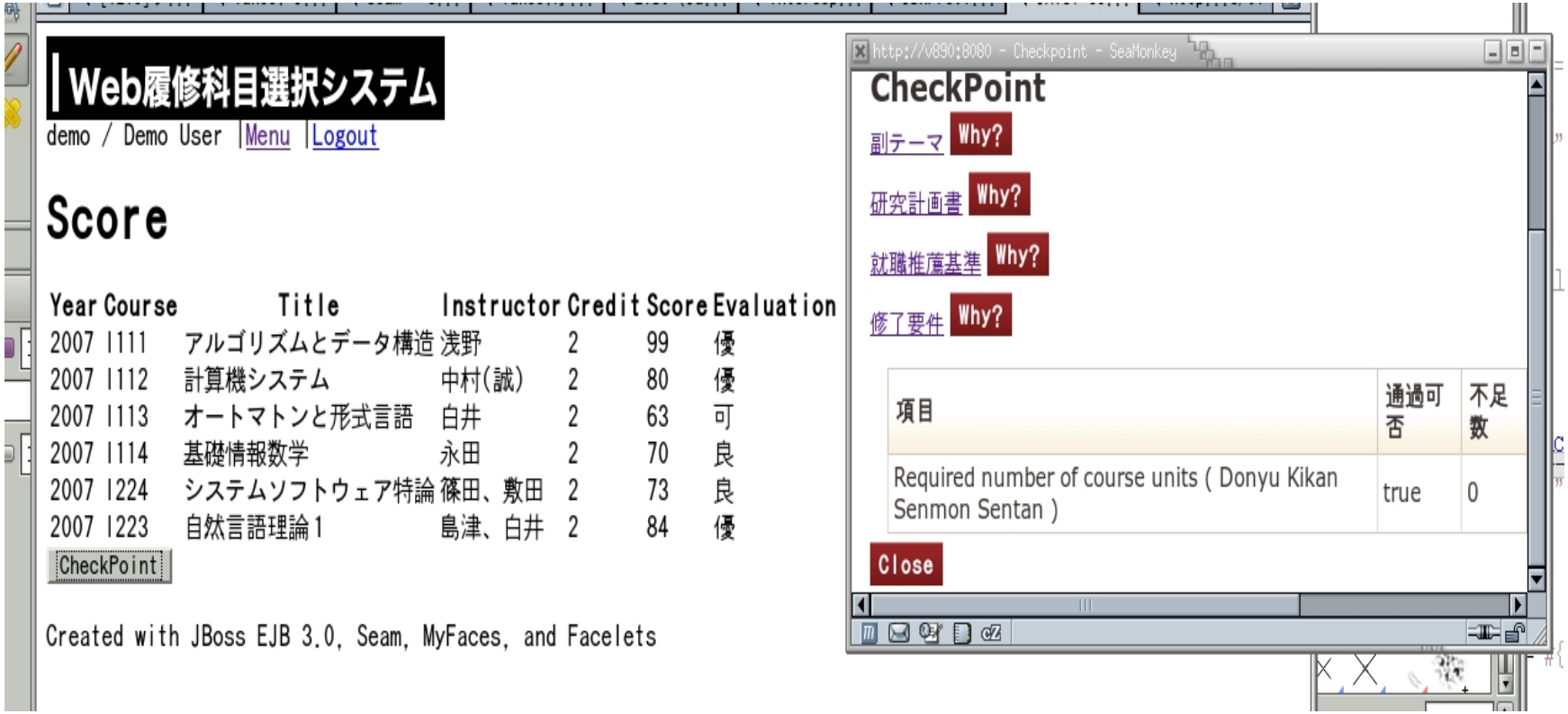
– JDIにより取得可能、ただし、実行時オーバーヘッドが特大

6. 質問と回答

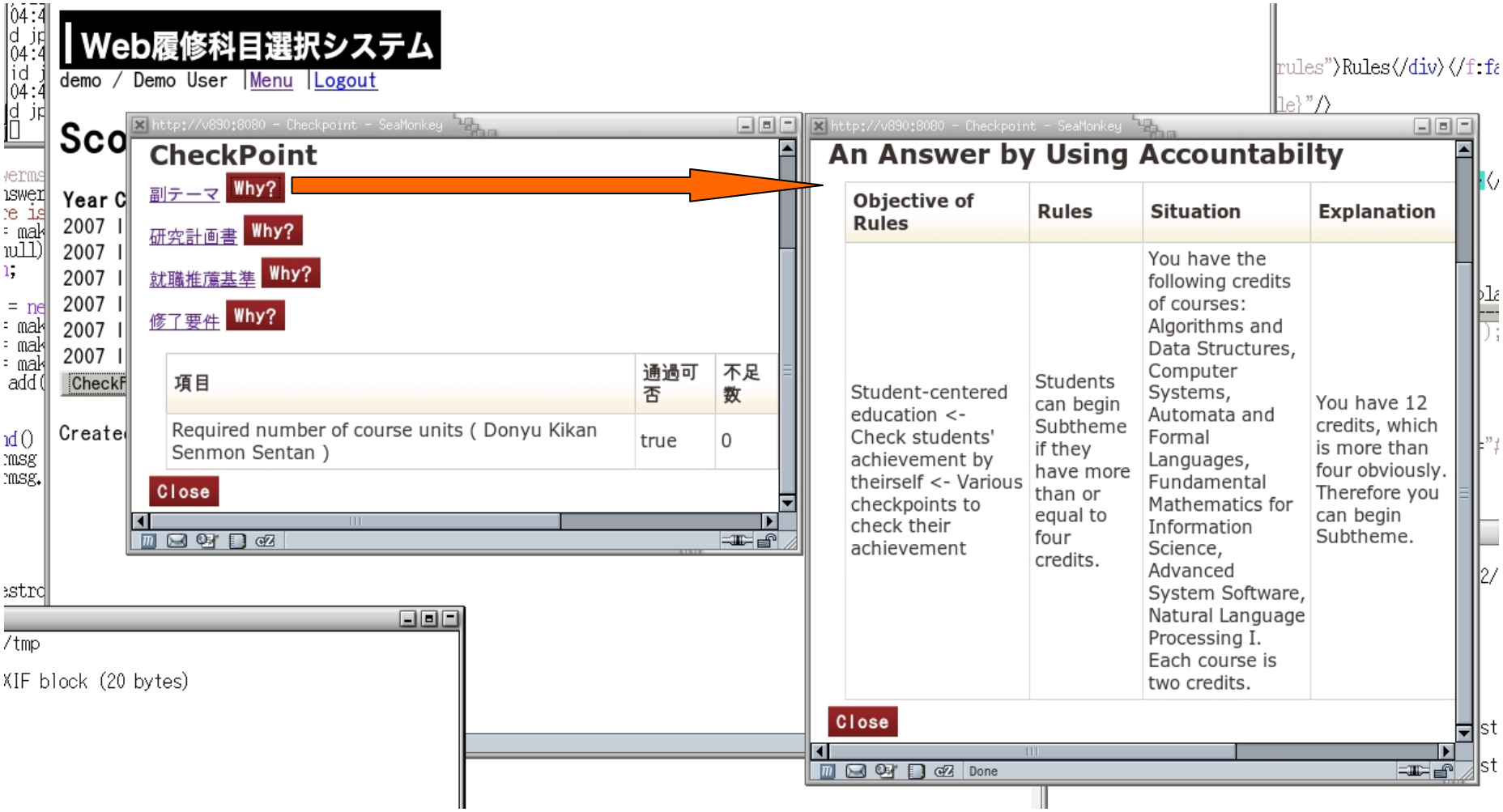
– 出力された回答に対する質問に対応する必要がある

# アカウンタビリティ機能に対する ユーザインタフェースの設計

- “Why?” ボタンを “Checkpont” ページに追加する
- 固定質問方式

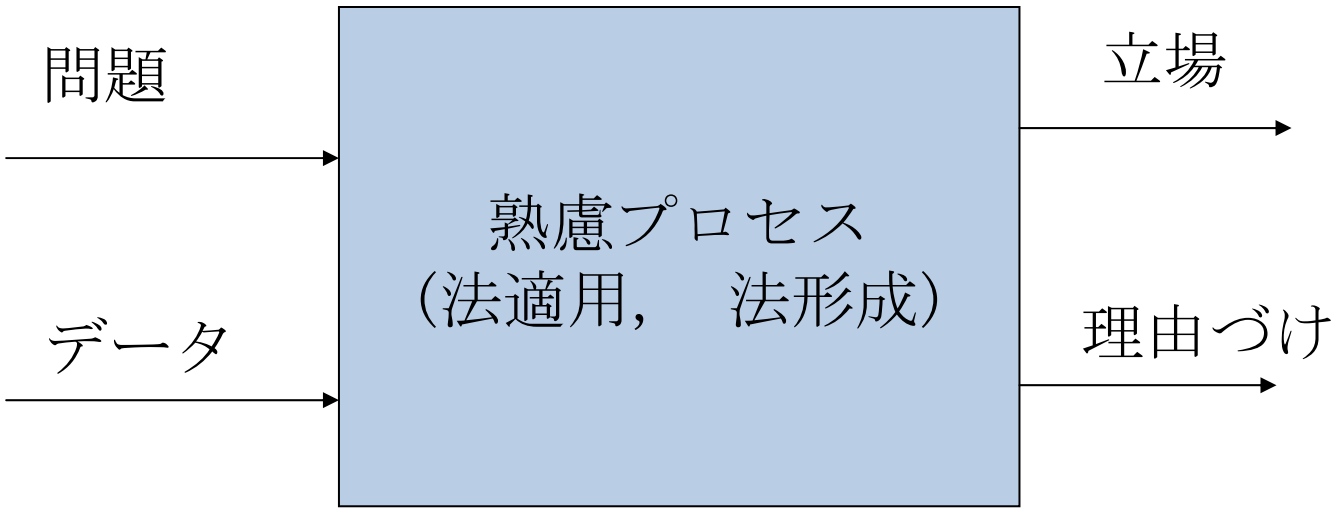


# 固定質問方式での回答生成例



# 課題

- アカウンタビリティモジュールの機能性の向上
  - インターセプタプロキシによって収集するデータの充実と性能問題の解決
- 進化容易性
  - 社会規則進化に応じたアカウンタビリティ木の版管理および属性の精密化

# 法システムにおける行為

- 法適用と法形成過程における熟慮プロセスはわれわれの関心に強い関係をもつ