ソフトウェア工学 85－7
（1992. 5. 21）

プロセス分割による協調的仕様化作業

西村一彦　本位田真一
(株) 東芝　システム・ソフトウェア技術研究所

本稿では、協調的なソフトウェア仕様化作業の自動化を実現するためのメカニズムを提案する。従来より提案されている仕様化技法を、その作業ステップの特徴を分類することによってプロセスモデルを作成する。そして、そのプロセスモデルに従って、作業プロセスを部分作業に分割する。本稿では, OOA[1],JSD[2],FOREST[3] の3つの技法に適用し, 作業プロセスの分割可能性について議論する. さらに, 2種類の部分作業間の協調メカニズムを提案し、プラン生成技術の応用を検討する.

A Mechanism for Implementing Cooperative Software Specification - A Basis for Partitioning Specification Process -

Kazuhiko Nishimura and Shinichi Honiden
Systems and Software Engineering Laboratory, Toshiba Corporation
Yanagi-cho 70, Saiwai-ku, Kawasaki, 210, Japan

This paper proposes a mechanism for implementing an automated cooperative software specification task. The specification process model is made from classifying the activities defined in the conventional method, which is done based on the multiple viewpoints analysis. The specification process is partitioned into appropriate sets of activities according to the specification process model. OOA, JSD and FOREST specification methods are partitioned and the results are presented. Afterward, two types of the cooperative mechanism among the sets of activities are proposed. The application of the planning technique for implementing the mechanism is discussed. The future research directions is also briefly presented.

1 はじめに

仕様化作業は，立場や専門知識の異なる人間（顧客と分析者，設計者など）が複数介在した協調作業の形態をなす [4]. この形態により，仕様化対象（実世界）を多くの観点から詳細にすることで作業結果の品質を向上する，作業を分担することで作業時間を短縮する、仕様化対象や作業環境の変化に柔軟に対処できるなどの利点をもたらす。そこで、近年、この協調作業の自動化や知的支援 [8] を実現するために、協調メカニズムを計算モデルとして確立することが要請されつつある。

協調メカニズムの研究は特に新しい分野ではなく、人工知能における協調分散問題解決 (CDPS)[7] とよばれる研究分野において様々な展開がなされている。だが、CDPS は分散化されたエージェント（部分問題解決器）間の協調メカニズムやプロトコルの領域独立な定式化に重点を置いた研究分野である．ここで、CDPS は個々のエージェントの記述に関しては比較的容易に行なえるという前提があるが、我々の対象とする仕様化作業のように，対象世界や作業環境の完全な定式化が困難な問題に対して，CDPS のアプローチを即時に適用できる可能性は低い。特に、協調メカニズムの定式化の前に，対象世界や作業環境の変化に対して柔軟なエージェントの構成を考える必要がある [7, 8].

本稿では、以上の背景から、エージェントの構成とエージェント間の協調メカニズムを提案する。具体的には以下の 2 点を明らかにする。

(1) 作業の分割の基準（作業の性質の違いを分類する）

(2) 作業間の協調メカニズム（作業間の相互関係のタイプを分類する）

2 章では、本稿で提案する協調システムの基本方針を述べる．3 章では，第一の問題である仕様化作業の分割方法の概要を述べ，既存の技法に対して適用する．4 章では，第二の問題である作業間の相互関係を明らかにし，協調作業のメカニズムを設計する上での重要なポイントについて述べる．

2 協調作業のモデル

1 章で述べたように、協調的な仕様化作業を行なうシステムを構築する上で、エージェントの役割を規定し、どのような知識を持ち、どのような推論（問題解決）を行なうかを明確にすることは非常に重要である。

理想的には、エージェントは仕様化作業全体に関しての知識や情報を各々の観点で持っており、お互いに協力して仕様化を行なう形態が望ましい。だが、システムの実現を考慮した場合、この理想論は作業プロセス全体に対する膨大な知識を記述しなければならない、エージェント間で一貫した知識記述を行なうことは稀であり、知識管理が困難である、エージェント間のインタラクションに関して、タイミングや情報の定式化が難しいといった問題を含んでいる。

本稿では、エージェントは仕様化作業の部分的な作業を行なうものとする。具体的には、ある技法が規定する作業プロセスを幾つかの部分作業に分割し、各エージェントに割り当てる。この作業分割の方法によれば，各エージェントは割り当てられた作業に関する知識や情報を持てば良い。そして、すべてではないにしても幾つかの部分作業は並行に行なうことができるので，全体作業の完了時間が早まる可能性が高い．例えば，OOA を大きく分けた場合，状態モデルとプロセスモデルの各作業はある程度作業が進むと，ほとんど並行して行なうことができる．さらに，同じ作業を異なる観点から遂行するので，最終的に得られる成果物は高品質な物を望むことができる．

3 プロセスモデルに基づく作業分割

既存の多くの技法は，作成すべき中間成果物の構成や意味的な違いから，仕様化作業をいくつかの大きなステップに分けている．例えば，OOA は情報モデル作成，状態モデル作成，プロセスモデル作成の 3 つの作業ステップに分割できる．実際の作業は，これらのステップを完全に同時並行的に進めることはできない．なぜならば，情報モデル作成の結果を利用して，状態モデル作成，プロセスモデル作成は作業を行なう．しかしながら，もっと詳細にプロセスを分析することによって，並行作業が可能な部分も明らかになってくる．先の OOA の例では，状態モデルにおける「オブジェクトの状態の決定」と「オブジェクトの特殊化」は単に作業結果だけでなく，作業の性質が異なる．従って，同時並行的に進めることができる．だが，むやみに作業を分けてしまうことは，無意味な作業間のインタラクションを増加させてしまう．

本章では，プロセスモデルに基づいた作業の性質を分類する方法を述べる．

3.1 仕様化プロセスモデル

技法が規定する作業プロセスを形式的に記述した仕様化プロセスモデルによって，実世界の特性に応じたプロセスの選択や，ツールの統合，異なる技法間での情報の共有などを行なうことができる．これまでに，様々なプロセスモデルが提案されている．我々は，技法が規定する作業ステップの観点をベースとしたプロセスモデルを作成した [9]. この観点は，基本的に 4W1H (Who, Where, What, When, How) に関連した 3 つの側面から構成されている (図 1). 各技法の規定する作業は，最下層の単位 (activity) まで詳細化し，それぞれのアクティビティがどの観点と関連するかを整理する．また，同時に，アクティビティ間の推移関係 (アクティビティ間の情報の流れ) を記録する．そして，異なる側面における各観点間の類似度を計算し，観点間の相互依存関係を求める．詳細は [9] を参照されたい．

(1) 構成的側面 (Structural perspective)

Who や *Where* を示す実世界の構成的な側面は，個々の要素に対する観点 (Absolute)，要素間の関係に対する観点 (Relative)，そして，両者に関係する観点 (MAR) に分かれる．

(2) 機能的側面 (Functional perspective)

What を示す実世界の機能的側面は，静的 (Static) と，動的 (Dynamic)，中間 (MSD) に分類される．Static とは実世界の「対象」とその構成や特徴といった構成的な面を明らかにし，Dynamic は対象の「振舞い」の面を明らかにする．

(3) 行動的側面 (Behavioral perspective)

When や *How* を示す行動的な側面は，瞬間的な出来事，すなわち，時点をベースとして実世界を認識する (Transient) 場合と，不変的事実，すなわち，時区間をベースとして実世界を認識する (Continuous) 場合，そして，両方の性質を持つ MTC に分かれる．

以下，本稿では，各側面の相対する観点 (Absolute, Relative, Static, Dynamic, Transient, Continuous) を基本観点と呼び，その他を中間観点と呼ぶ．

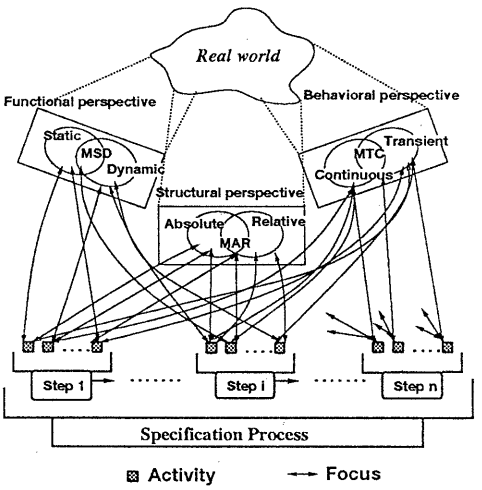

図 1: 多視点に基づいた仕様化作業

3.2 分割基準

前節で定義した 9 つの観点は相互に独立な概念ではある．従って，これらの観点と作業との関係が整理することによって，最大 9 つの作業分割が可能である．この分割方法のメリットは 2.2 で述べた評価基準を満足する．すなわち，その作業を行なうエージェントは対応する観点に関する情報 (知識) のみを持てば良い (評価基準 b))．そして，作業間のコミュニケーション情報は観点間に流れる情報によって整理でき (評価基準 c))，独立な観点に対する作業は並行に行なえる可能性が高いと考えられる (評価基準 a))．さらに，本方法は，同じ作業プロセスを複数の側面から分類しているので，一つの作業は多重に行なわれる (評価基準 d))．

本稿では，各技法の規定するアクティビティと 3.1 で定義した 9 つの観点 (3 つの側面) 間の対応関係をベースとした観点に基づいた作業分割の可能性を以下のように提案し，既存の技法の分割を行なう．

(1) 各側面ごとに，次の分析を行なう．あるステップ $Step_i$ (アクティビティの集合) が，特定の観点 V に十分に依存している場合は，ステップ $Step_i$ と観点 V は関連があるものとする．ここで，$Step_i$ が V に十分に依存するとは，以下の条件を満足する：a) $Step_i$ のすべてのアクティビティが，V だけに属している．あるいは，b) アクティビティがどちらか一方の基本観点と中間観点に属しているならば，アクティビティの属している数が多い観点に対して依存するものとする．アクティビティが同数の場合や対抗する基本観点に属している場合は成立しない．例えば，図 3 の左図で，ステップ C({C1, C2, C3}) はすべて Relative に属しているので，ステップ C は Relative に十分依存する．また，ステップ A({A1, A2, A3, A4}) は，$A4$ が MAR に属しているが，条件 b) を満足するので，Absolute に十分依存している．しかしながら，ステップ B や D はこの条件を満たさない．

(2) 3.1 で定義した 9 つの観点間の相互依存関係情報をもとにして，類似する観点をまとめる．そして，(1) の結果を修正する．すなわち，ある作業ステップが複数の観点に対応している場合，それらの観点間の類似度が 0 でない場合は，それらをグループ化する．

3.3 既存技法の作業分割

図 3 – 図 5 は，OOA, JSD, FOREST の各技法のアクティビティを分類した結果である．図中，楕円で囲まれたアクティビティは対応する観点に十分に依存するものである．また，アクティビティ間の情報の流れにおいて，黒丸はアクティビティ間の情報の合流を表し，白丸は情報の分岐を表す．

図 6 は，分類結果に対して観点 V_a, V_b 間の類似度を式 (1) によって計算した結果をまとめたものである．

$$S(V_a, V_b) = \frac{size(V_a \bigcap V_b)}{size(V_a \bigcup V_b)} \qquad (1)$$

表 4 は，上記 (1)-(2) の分析を行なった結果であり，観点と作業ステップとの関係を示している．

3.4 各技法の特徴

(1) OOA

OOAの6つの作業ステップは，一つ以上の観点に対応している．すなわち，作業ステップ毎の作業分割が可能であることを示している．また，観点の相互依存関係の強いものが多く，幾つかの作業は複数の観点に対応する．例えば，オブジェクト記述は(Static, Absolute, MTC)に対応している．

(2) JSD

JSDは，他の2つの技法に比べて，作業ステップと観点との対応関係は少ない．初期モデル段階が3つの観点に対応してだけである．また，一部の作業ステップは，対抗する基本観点を持つという特徴がある．例えば，図4の機能的側面において，ステップAはStatic, MSD, Dynamicに分かれている．従って，観点に基づいて作業ステップを分割することは困難である．

(3) FOREST

FORESTは，OOAと同様に作業ステップと観点の対応関係が顕著であり，観点に基づいて作業を分割できる可能性が高い．特に，データフロー分析は3つの観点に対応している．特徴的なことは，OOAと異なり，一つ作業が複数の観点に関連している場合，その観点間の類似度は高く，OOA以上に分割が容易である．

4 協調作業のモデル

前節において，既存技法に対して，観点をベースとした作業分割の可能性を分析した結果，JSDを除く2つの技法は，作業分割が容易であることが分かった．本章では，この分析結果から協調作業を効率良く行なうための作業間の相互関係について考察する．

4.1 エージェント

エージェントは分割された作業を遂行する問題解決器であり，目的（e.g. オブジェクトの分析）が与えられるとその達成手順（e.g. インスタンスの識別）を決定するとともに，必要に応じて，他のエージェントと協力することによって全体作業を行なう．例えば，OOAにおいて，オブジェクト間の関係を分析するエージェントは，属性記述を行なうエージェントの作業結果である各々のオブジェクトの属性を必要とする．また，仕様化作業は，各々のエージェントで得られた結果が矛盾を含んだり，不完全な状況が頻繁に発生する．従って，矛盾の原因を探るために，矛盾を起こした作業過程を求めたり，さらに，不完全な部分を補うための作業手順を求めることが必要となる．この問題は，ＡＩプランニングの問題と共通の性質を持っており，部分的にはその成果を利用できると考えられる[10].

エージェントのタイプとして，「タスク調整タイプ(C-Agent)」と「タスク遂行タイプ(A-Agent)」を仮定する．

「タスク調整タイプ」は，3.1で定義した3つの側面（構成，機能，時間）に対応するエージェントであり，各側面毎に行なわれている仕様化作業の調整を行なう．例えば，構成的側面で明らかになる情報を機能的側面の情報とマージするタイミングを決定したり，ある側面で問題が発生した場合に，他の側面とどのようなに協調すべきかを決定する．

「タスク遂行タイプ」は，各側面の3つの観点に対応するエージェントであり，実際の仕様化作業を遂行する．すなわち，作業手順の決定とその遂行である．

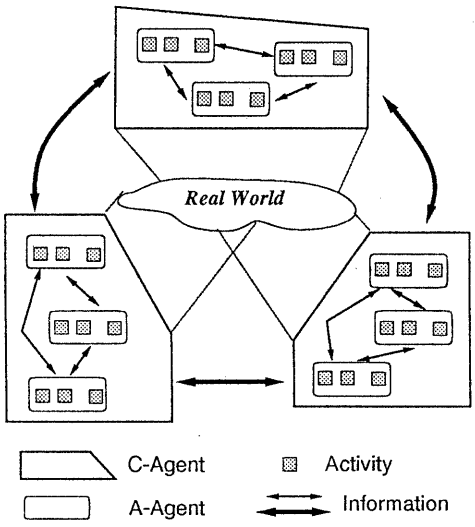

図 2: 協調作業のモデル

4.2 協調メカニズム

(1)「タスク調整タイプエージェント」の協調メカニズム

1章で述べた様に，仕様化作業は実世界を複合的に分析する作業である．複合的に行なう理由として，成果物の品質向上がある．例えば，OOAでは，オブジェクトの属性は，オブジェクトの構成やオブジェクト間の関係のみから明らかになるのではなく，そのオブジェクトの振舞い（状態属性）からも得られる．従って，側面毎に行なわれる作業はある点でマージすることが必要となる．だが，むやみにマージすることは，意味のない相互干渉を引き起こし，作業効率の低下を招くことになる．そこで，本稿ではマージポイントとして，観点間の類似度(図6)を利用する．すなわち，各側面で行なう作業は，観点間の類似度が高い場合には，互いに協力しあって（すなわち，同期を取る）作業を行ない，類似度が低い場合には，各々独立に作業を行なう．これにより，マージ

ポイントが明確になり，全体の作業効率も高くなると予想される．

(2)「タスク遂行タイプエージェント」の協調メカニズム

図3等で明らかなように，各観点の作業は観点ごとに独立なものではなく，お互いに，他の観点の作業結果を利用しあっている．例えば，OOAでは，Absoluteエージェントによって明らかになったインスタンス名(B2の出力)をもとにして，Relativeエージェントが B4を行なう．また，このように別個の観点で同じ作業（この場合オブジェクトの属性記述）を行なった場合，各々の観点で得られた作業結果の調整を図る必要性も出てくる．例として，簡単な図書館システムの仕様化を考える．「本」というオブジェクトに対する属性として，「タイトル」「著者」「ISBN」はAbsoluteな観点から得られる．一方，Relativeな観点では「図書館のユーザ」と「本」との関係から，その本を借りているユーザの「ユーザID」という属性が得られる．これは，得られた結果が互いに補間しあっている例である．別な例としては，「本」が借りられている期間は「本」の属性として定義しても良いし，その本を借りている「ユーザ」の属性としても良いので，どちらか一方を削除することが必要である．

こうした協調作業のポイントとなる部分は，観点間に流れている情報の合流点と分岐点である(図3-図5の黒丸と白丸)．すなわち，必要な情報を適切な時期に得るための制御メカニズムをこれらのポイントに流れる情報の特徴によって設定することが必要である．

5 まとめ

本稿では，協調的な仕様化作業の知的計算機支援を実現するために，エージェントの構成とそれらの協調メカニズムを提案した。

本方法は、従来より提案されている仕様化技法を対象世界の独立な複数観点(「誰が，いつ，どこで，何を，どのように」)と作業ステップとの関連に基づいたプロセスモデルに従っている．本稿では，OOA, JSD, FORESTの3つの技法に適用し，各々の技法の規定する作業の分割の可能性について議論した．その結果，OOA, FORESTは観点をベースとした作業分割が可能であり，JSDは観点をベースとする作業分割が困難であるという結論を得た．

さらに，この分析結果をもとに，協調的な仕様化作業を行なうエージェントの構成とエージェント間の協調メカニズムを明らかにし，プラン生成技術の応用について検討した．

しかしながら，ＡＩプランニングにおける初期状態，目標，オペレータに相当するものが何かを明確にすることが必要である．特に，仕様化作業の場合，対象世界を完全に規定することは困難であり，不完全な世界をどのように表現し，また，不完全さから起きる種々の問題の対処方法を明確にしなければならない．

今後は，協調メカニズムをインプリメントし，定量的な評価を行なう予定である．

参考文献

[1] S. Shlaer and S. J. Mellor : *Object-Oriented Systems Analysis Modeling the World in Data*, Prentice-Hall Inc, 1988. (本位田, 山口訳：オブジェクト指向システム分析 - 上流 CASE のためのモデル化手法 -, 啓学出版, 1990).

[2] M. Jackson : *System Development*, Prentice-Hall Inc., 1983. (大野, 山崎訳：システム開発 -JSD 法 -, 共立出版,1989).

[3] A. Finkelstein and C. Potts : Building Formal Specifications using Structured Common Sense, *IWSSD*, pp. 108–113, 1987.

[4] A. Finkelstein and H. Fuks : A Cooperative Framework for Software Engineering, *ICSE*, pp. 189–199, 1989.

[5] S. Fickas and P. Nagarajan : Being Suspicious: Critiquing Problem Specifications, *AAAI*, pp. 19–24, 1988.

[6] W. N. Robinson : Integrating multiple specifications using domain goals, *IWSSD*, pp. 219–226, 1989.

[7] E. H. Durfee, V. R. Lesser, and D. D. Corkill : *Cooperative Distributed Problem Solving*, Vol. 4, chapter 17. Addison Wesley, 1989.

[8] S. Fickas and R. Helm : Acting Responsibility: Reasoning about Agents in a Multi-agent System, Technical Report CIS-TR-91-02, Department of Computer and Information Science University of Oregon, 1991.

[9] 西村一彦, 本位田真一： 複合ビューポイントに基づく仕様化プロセスの定性的分析. 情報処理学会ソフトウェア工学研究会, Vol. 81-6, pp. 41–48, 1991.

[10] J. S. Anderson and S. Fickas : A Proposed Perspective Shift: Viewing Specification Design as a Planning Problem, *IWSSD*, pp. 177–184, 1989.

[11] 西村一彦, 本位田真一： 協調的仕様化作業のためのプロセス分割可能性の検討, 第4回ソフトウェアプロセスワークショップ, 1992.

表 2: OOA の仕様化プロセス

Step	Objectives	Activity
A	Define objects	1.Identify instances 2.Describe objects 3.Name objects 4.Examine object definition
B	Classify attribute categories Determine identifications Record the values of each attribute	1.Describe fundamental features of instances 2.Define instance name or label 3.Define connection of instances 4.Define identification independency 5.Set attributes according to pre-defined category
C	Describe binary relations	1.Model relations with no conditions 2.Model relations with conditions 3.Simplify M:M relations
D	Specialize or Generalize Make relative objects	1.Incorporate all attributes except the identification 2.Relations with instances
E	Determine the behavior of each objects Add the state attributes	1.Define events 2.Define states and actions of each object 3.All states in life cycle
F	Identify and describe actions	1.Define data flow from attributes 2.Define data store 3.Define actions as processes

表 3: JSD の仕様化プロセス

Step	Objectives	Activity
A	Identify actions List attributes Identity entities	1.List of verbs 2.Select the immediately and primitive actions 3.Resolve the conflicts by taking notice of common actions 4.Identify the attribute of each action 5.List nouns 6.Think that they can do and affect the described actions 7.Form the entity types by generalizing and abstraction 8.Name by distinct label
B	Arrange actions in their ordering by time Refine the entity structure	1.Consider whole life span of each entity 2.Describe the order which the action can happen 3.Take notice of concurrency [Add marsupial entities] 4.Specify the separate structure for each role 5.Check composite structure
C	Realize the entities and actions in a process and connections between the model and real world	1.Make the model processes 2.Connect the real world to the model 3.Merge multiple inputs stream
D	Add functions which generate required outputs Refine the model processes	1.Determine what functions are needed to generate the outputs to the model 2.Establish connections between function processes and model processes 3.Add interactive functions to generate inputs for internally generated actions 4.Introduce the time processes into the model
E	Consider some aspects of process scheduling	1.Consider potential delays which affect to function output 2.Add synchronization processes into SSD

表 4: FOREST の仕様化プロセス

Step	Objectives	Activity
A	Identifying agents Describe the agents	1.Identify the agents relevant to the system be specified 2.Construct the hierarchy of agents 3.Elaborate the agents' description
B	Analyze data flow	1.Analyze data flow between agents 2.Check the consistency of agents' interfaces
C	Identify actions performed by each agent	1.Model what the agent can do 2.Check the production and consumption of data flows 3.Consider internal data flow 4.Constraint the co-occurrence of input and output data flows 5.Describe in natural language
D	Identify data sort and add structure to it Identify static structure of the system	1.Decompose data flows 2.Examination of the agent hierarchy, data structure and action description 3.Model the passive entities 4.Identify the attributes which are tied to specific entities 5.Classify the relationships
E	Analyze permission and obligation Make causal table	1.Consider condition under which each action may, can and must occur 2.Consider what enables each action, what disables it and what effects it has 3.Signal data flow between the two actions 4.Tabulate the causal relations between two actions

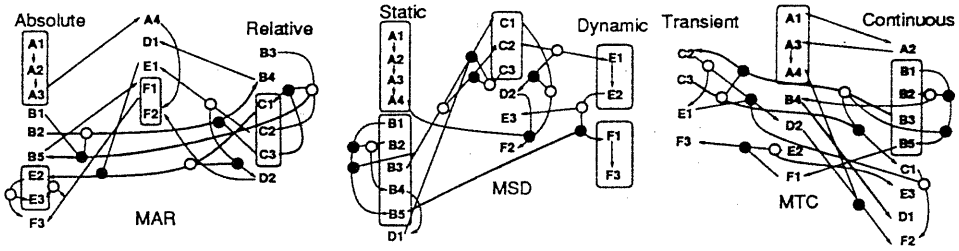

図 3: OOA のプロセスモデル

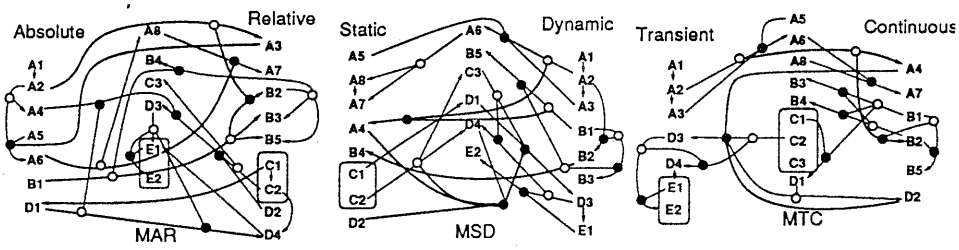

図 4: JSD のプロセスモデル

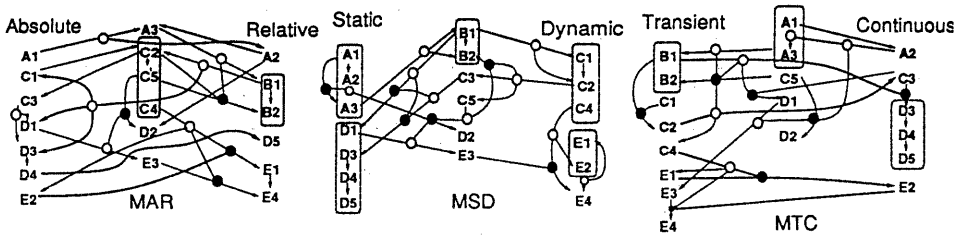

図 5: FOREST のプロセスモデル

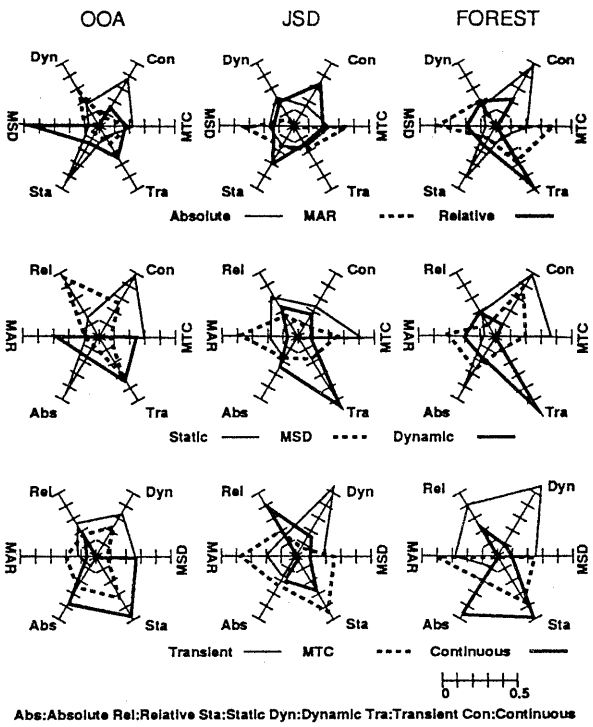

図 6: 観点間の類似度

表 5: 各技法の仕様化プロセスの分割

	観点	作業ステップ
O	Absolute, Static, MTC	オブジェクト記述
O	Static, Continuous	属性記述
A	Relative, MSD	関係記述
	Dynamic	状態モデル，プロセスモデル
	Absolute	状態モデル
	MAR	プロセスモデル
J	Relative, Static, MTC	初期モデル段階
S	MAR, Transient	システムタイミング段階
D		
F	Static	動作主識別
O	Relative, MSD, Transient	データフロー分析
R	MAR, Dynamic	行動分析
E	Static	実体分析
S	Dynamic	因果分析
T		