人 工 知 能 77−10
（1991. 7. 18）

協調アーキテクチャへのアプローチ

中島秀之
電子技術総合研究所 協調アーキテクチャ計画室

特定の能力をもつプログラム単位を多数集めることにより，それらの能力の総和以上の働きをさせることが可能になるような，新しい構成手法を考える．これを協調アーキテクチャと呼ぶことにする．つまり，多数のエージェントが協調することにより，知的で複雑な作業が可能なようにするアーキテクチャである．本論文では，この協調アーキテクチャへのアプローチを，状況推論の考え方を中心に示す．

Approaches to Cooperative Architecture

Hideyuki Nakashima
Cooperative Architecture Section
Electrotechnical Laboratories

We are now developing a new software methodology for building large complicated systems out of simple units. The emphasis is on the architecture to combine the units, which we will call “Cooperative architecture”. The architecture enables cooperation of units to achieve complicated intelligent tasks. One of the approaches, based on the idea of situated inference is introduced in this paper.

1 はじめに

高度な知的システムを実現するのに従来の手法では限界が見えており，新しい手法が要求されている．特定の能力をもつプログラム単位（以下エージェントと呼ぶ）を多数集めることにより，それらの能力の総和以上の働きをさせることが可能になるような，新しい構成手法（アーキテクチャ）を考える．これを協調アーキテクチャと呼ぶことにする．つまり，多数のエージェントが協調することにより，知的で複雑な作業が可能なようにするアーキテクチャである．

従来の手法との差を考えてみよう．たとえば，従来のソフトウェア工学では，プログラムの部品化を考えてきた．これはプログラムの動作を規格化し，どこへ持って行っても同じ動きをすることを目的としている．これに対し，我々はより高度なソフトウェアを開発する観点から，柔軟なシステムを求めている．あるプログラムが環境に適応し，適当に動作を変えることを目的としており，これは従来のソフトウェア工学の逆の発想である．イメージ的に言うと，従来の，ボルトやナット（あるいはエンジン）のような共通部品ではなく，細胞のような共通部品を求めているのである．

問題解決においても，従来“分割統治”の名に示されるように，問題をより単純な部分問題に分割して解決するという手法がとられてきた．しかしながら，各々が（完全ではないにしても，ほぼ）独立な部分問題に分割できるような問題 (nearly decomposable problem[NS72]) はその複雑さに限度があり，ここで扱おうとする高度な問題はそういう性質を持っていない．すなわち，分割された部分問題は相互に依存しており，これらの関係をうまく扱わない限り全体は解決できない．部分間の協調が必要とされるのである．我々の目標はこのような複雑な問題を定式化する考え方の設立にある．

ここで使う言葉の定義をしておく．

エージェント 特定の能力をもつプログラム単位．モジュールの概念に近いが，情報隠蔽は含意しない．また，個々のエージェントが知的である必要はない．[1]

アーキテクチャ システムの構成手法．特に全エージェント間の相互作用方式のこと．従って，（オブジェクト指向や論理プログラミングと並ぶ／越える）新しい計算パラダイムとなるものである．

協調 多数のエージェントによる整合的な営み．協力，折衝，妥協などを含む一般的相互作用のこと．

2 研究の目的

我々の研究の目的は特定の能力をもつエージェントを多数集めることにより，それらの能力の総和以上の働きをさせることが可能になるような，協調アーキテクチャを創出することにある．

2.1 並列・分散・協調

“並列”，“分散”は処理の形態のこと，“協調”は処理の目的あるいは処理の方式のことである．我々が対象とするのは並列／分散環境において動作するエージェントの協調方式であるが，特に並列や分散を表だって扱うことはしない．

分散人工知能と共通するのは情報や処理の局所性を前提とすることである．しかしながら，両者は，分散人工知能が，ある問題を分散環境で如何にして解くかを問題にしているのに対し，

[1]問題の複雑さに応じ，個々のエージェントにも知能が要求されることを否定しているのではない．全体の知能を個々の知能に還元するという（循環論的）方法論を採らないという意味で解釈されたい．

協調アーキテクチャが多くのエージェントを構成してより複雑な問題を解くことを問題とする点において異なる．つまり，前者は問題が処理の中心となるトップダウン方式，後者はエージェントが構成基準となるボトムアップ方式と言えよう．特に，分散人工知能において，分散というのは大前提であり，たとえそれが否定的要因であっても取り除くことができない．それに対し，協調アーキテクチャでは肯定的な場面でのみ，必要とされる並列／分散性を取り込むことになる．

2.2 三人よれば文珠の知恵

協調により，個々のエージェントの能力の和以上の仕事をさせることを目標とする．並列計算においては，個々の計算機の単位時間あたりの計算能力をPとするときに，一般にn台の並列計算でnPの計算をこなせることはまれである．もちろん，これは計算の内容に依存しており，各々のプロセッサが完全に局所的に（例えば2次元格子状に配置されているプロセッサでは4方向の隣のプロセッサとのやりとりだけで計算が済む）計算を進めることのできるような問題ではnP オーダーの計算が可能であるが，全体的な通信を必要とするような問題では，通信に log(n) 時間かかるとして，log(n)P あたりが現実的な線であろう．

計算量を評価基準とすれば，上記のように全体のパフォーマンスは個体の和を上回ることは困難であるが，解ける問題のクラス等を評価基準とすれば，個体では解けなくとも全体で解けるようなクラスが存在するはずである．計算の例でなくて恐縮であるが，例えば重いものを持ち上げる場合に，一人のエージェントが片端づつ時分割で持ち上げても無駄である．三賢者の問題[NPS91]などは好例とならないだろうか？三人よれば文珠の知恵である．

3 協調アーキテクチャへのアプローチ

我々は協調アーキテクチャに対し，以下のような枠組から取り組む予定である．

3.1 状況推論

エージェントとその環境の関わり合いを考えることにより，エージェントに柔軟な振舞いをさせることが可能となる．また，自然言語の持つ状況依存性を研究することはエージェント間の通信の問題を解決する糸口になると考えている．詳しくは4節で述べる．

3.2 オブジェクトオリエンテドな考え方の拡張

システムを小さなオブジェクトの集まりに分解し，それらの協調として全体をみるという見方はオブジェクトオリエンテドの考え方そのものである．しかし，現在のそれはCLU[Lea79]等の抽象データ型の考え方の発展として考えられており，分割統治の考え方に近いものである．

ひとつの問題は，メッセージパッシングの考え方が，オブジェクト間の協調よりも，一方的通達に近い点にある．これを改善するには，メッセージパッシングよりも広いバンド幅の通信を考え，相手の内部状態等が観察できるようにする[HS86]，メッセージの送出と受信を分ける[Gel85]，メッセージ交換をプリミティウ゛にとる[FF86]等の方向が考えられる．しかしながらいずれも現状では満足のいく解ではない．

3.3 リフレクション

人間の環境適応能力の一部は，自分を外側からとらえるリフレクションの機能によると考えられる．リフレクションにより，エージェントと環境の関係を客観的にとらえ，環境に応じてエージェントを変更することが可能となる．

3.4 共同行為の理論

共同作業に関して，共同目的，共同意図などをモデル化する（例えば[CLNO90]）ことも重要である．また，エージェント間で意図や利害が対立する場合の折衝や妥協の理論も必要である．

3.5 高階タイプ理論

複雑なシステムは常に変化する．システムに対する要求仕様が変化することもまれではない．協調システムにおいては，一部の仕様変更が全体に影響を与えないように，それを局所的におさえこむ必要がある．

オブジェクトオリエンテド言語ではオブジェクトクラスの増加に関してはその新しいクラスに関する記述を追加するだけでよい．つまり，クラスの増加による変更が局所的におさまる．しかしながら，新しい種類のメッセージを追加する場合には，関連する全クラスを変更する必要が生じ，この変更は大局的になる．

仕事が変わっても，メッセージが変わっても，それによって変更を受けないもの（たとえば“意図”）をプリミティヴにとり，それを用いて仕事の内容やメッセージの意味を記述することにより上記の要請（変更に対する局所性）が満足できる可能性がある．その候補として，例えば“メッセージとは何か”を記述できるような高階記述言語が考えられる．高階タイプ理論[Gog87]はその足掛かりとして期待されている（応用に関しては，例えば[GMP82]）．

4 状況推論からのアプローチ

システムが“知的”に振舞うためには，そのシステムが状況を把握し，それに対して適切な行動をとる能力を有する必要がある．しかし，状況を把握していなくても適切な行動をとることは可能である．例えば，自動販売機は挿入された硬貨を識別し，それに応じて適切な行動をとる．しかし，硬貨の直径や重さにより硬貨を弁別しているため，偽造硬貨に対しては対処しきれないし，製品の価格変動にも自らは対応できない．すなわち自動販売機は設計された範囲内では適切な行動をとるが，そこからはずれると対応しきれない．動物行動学でも動物の同様な環境依存が示されている[テ 75]．“知的”な行動とはこのような予期された状況からの逸脱に対する適切な（あるいは可能な限りの）対応が含まれると考えられる．

状況理論では，従来，自動販売機のような状況に対する適応（attunement）しか考慮されなかったが，ここでは“知的”行動のための状況の把握やその利用を考える．そのために，“状況に関する推論”[NT91]の必要性を提唱する．なお，状況理論の考え方を出発点にしてはいるが，ここでの考え方は筆者独自のもので，状況理論の開発者たちとは立場の異なる部分が多い．もっとも大きな相違点は，彼らが状況は推論者の外にあると考えているのに対し，ここでは状況の内部表現を考え，状況を思考の対象としている点である．

状況とのかかわり方について以下の二つを考える：

1. 状況への同調

   システムが“知的”に振舞うためには，そのシステムが状況を把握し，それに対して適切な行動をとる能力が必要である．しかし，状況を完全に把握していなくても適切な行動をとることは可能である．例えば，自動販売機は挿入された硬貨を識別し，それに応じて適切な行動をとる．この場合，適切な行動をとるのに必要な情報をすべてシステムが持っているのではなく，システムをとりまく状況内にある情報（コインの直径や重さと金額の関係など）に依存している．状況内オートマタ[Ros87]と同じ考え方である．

2. 状況の扱い

自動販売機は，設計された範囲内では適切な行動をとるが，そこからはずれると対応しきれない．単に状況に同調しているだけではなく，状況との積極的なかかわりが要求される．これには，状況からの情報の収集や状況の切替えなどが含まれる．

エージェント間の通信を考えるうえで，自然言語がよい手本となる．しかも，それに留まらず，エージェント内での推論システム設計の指針ともなるというのが我々の作業仮説である．自然言語による通信では状況にある情報を最大限に利用している．つまり，我々が用いる言葉の内容はそれを使用する状況に依存する．言語の状況依存性には，

1. 実際に文には表れてこない情報（時間や場所の情報など）が内容には含まれること．(unarticulated constituent.[Bar89])

2. 単語とその内容とのマッピングが状況に依存すること．

の二つがある．ここでは，このような二つの状況依存性が自然言語だけの特徴ではなく，思考の言語，つまり知識表現やその上の推論機構にも見られると考える．この依存性を積極的に操作することが可能であると考えることにより，表現の柔軟性，推論の効率化などが可能になる．これが状況推論である．

4.1 解釈と視点

ある単語とその指示対象（内容）の関係を解釈と呼ぶことにする．この逆の関係が視点である．後者はある概念をどういう目でみるか？どういう単語で表現するか？という選択である．例えば，ある色を赤というのか，紅というのかの選択；ある動物をシェパードというのか，犬というのか，動物というのかの選択，等がそうである．視点は話し手，解釈は聞き手の選択である．両者が一致したときに初めて意味が伝達される．

思考の場合には思考言語のなかの単語に相当する単位に対してこれがみられる．思考の場合は同一人物が行なうので視点と解釈の不一致は普通は考えなくてよい．

しかし，ある単位（状況理論ではパラメータと考えてよい）が何を指すのかは状況依存である．それにより“私は喉が乾いた”という思考が，万人に対して同じ行動を惹き起こすことが可能となる．つまり，太郎が“私は喉が乾いた”思う時は太郎が自分で水を飲めばよいし，花子が“私は喉が乾いた”思う時は花子が自分で水を飲めばよい．そうではなくて“太郎は喉が乾いた”という思考だと，太郎と花子に対しては異なる行動を要求することになるので，うまくない．

つまり，太郎が“太郎は喉が乾いた”と思う時は自分で水を飲めばよいのだが，花子が“太郎は喉が乾いた”と思う時は太郎に水を飲ませると言う別のタイプの行動をとる必要がある．つまり，同じ思考が人によって異なる行動として表れることになる．

具体的な内容ではなく，状況に依存する表現をとることにより，状況を抽象化してとらえ，同じ推論やプランを異なる場面で使用することが可能になる．

4.2 投影

ある n 項関係を n-1 項関係に落とすこと．ここでは，状況にある情報を表現（思考，言語両方を含む）中の引数から落すことを指す．

Palo Alto で It's 4pm. と言う／考えることは，実際は Palo Alto が PST で 4pm であることを意味している．

⟪4pm, PST, Palo Alto⟫

が実際に仮定されている情報である．

もし，実際の発話が

⟪4pm⟫

（に対応する自然言語の）のかたちをしており，その発話の内容が

⟪4pm, PST⟫

だとすると，“PST”は発話には含まれないが情報の構成要素であるという意味で unarticulated constituent[Bar89] と呼ばれる．投影とはある constituent を unarticulated にする操作である．

ただし，投影前の表現は固定でも完全な情報を含んでいるわけでもない．完全な情報とは

⟪4pm, PST, Palo Alto, 太陽歴, ...⟫

のように，フレーム問題的に何が必要かわからない．多分，その状況（Fauconnier のメンタルスペース [Fau85] に近い）がサポートしている“関係”を全部並べたものがフル情報だろう．

従って，

⟪4pm⟫
⟪4pm, PST⟫
⟪4pm, Palo Alto⟫
⟪4pm, PST, Palo Alto⟫

等のすべてを実際のものとして認める．つまり，何らかの完全情報とそれを省略したものという対比をとらずに，すべてが実際にあるものとして考える．上記の例では（上が下の）投影の関係になっているが，必要に応じて元の情報を復元できる．それは，状況にその情報が含まれているからである．これらは（上が下の）投影の関係になっているが，必要に応じて元の情報を復元できる．それは，状況にその情報が含まれているからである．この情報がどのような形になっているかを考えてみる．PaloAlto という状況を考えよう．これは少なくともそこでの時刻表示が PST であるという情報をサポートしている：

PaloAlto $\models$ ⟪timezone, PST⟫

ここで

USA $\models$ ⟪4pm, PST⟫ $\Leftarrow$ PaloAlto $\models$ ⟪timezone, PST⟫$\wedge$ ⟪4pm⟫

のような制約を考えると

PaloAlto $\models$ ⟪timezone, PST⟫
PaloAlto $\models$ ⟪4pm⟫

と

USA $\models$ ⟪4pm, PST⟫

とは同じ情報を持っていると考えることができる．[2]

状況や，その上の情報の投影の詳しい理論は別稿 [NOK91] に譲る．

---

[2]状況理論では，

PaloAlto $\models$ ⟪4pm⟫

は Palo Alto で“本当に”／“実際に” 4pm であるということ．そうであるという表象（Palo Alto ではなく，頭の中にある）はサポートとは異なる関係であるが，ここでは同一視している．

4.3 前提条件の削除

前節では投影によるパラメータの省略を考えたが，ここでは前提条件の省略を考える（基本的には同じ現象だと考えている）．例えば

$\langle\langle$fly, Tweety$\rangle\rangle$

は

$\langle\langle$bird, Tweety$\rangle\rangle$
$\langle\langle$fly, X$\rangle\rangle \Leftarrow \langle\langle$bird, X$\rangle\rangle$

から導けるが，これらを省略する，効率の良い推論を考えることもできる．まず，

$\langle\langle$bird, X$\rangle\rangle$

が必ず成立するような状況のタイプを *bird* とする．つまり：

$$bird = [\dot{s} \mid \dot{s} \models \langle\langle \text{bird}, \dot{X} \rangle\rangle]$$

である．同様に Tweety という個体もこれを “Tweety である” という述語で表すことにし，

$\langle\langle$Tweety, X$\rangle\rangle$

が必ず成立するような状況のタイプを *Tweety* とする．そうすると，

$$bird \subseteq Tweety$$

により，推論を状況の包含関係で置き換えることが可能である．この包含関係は，状況タイプを，それがサポートするインフォンの集合として考えた場合のその集合の関係である．従って，包含する側のタイプの方が多くの情報をもち，そのタイプに属するインスタンスは少なくなる．つまり

$$\forall T_1, T_2. T_1 \subseteq T_2 \rightarrow (\forall x\ x : T_2 \rightarrow x : T_1)$$

が成立する．

ちなみに，上記の考え方はフレーム理論 [Min75] に発展させることができる．フレーム理論でははプロトタイプ的知識がフレームと呼ばれるあるまとまった形式で蓄えられていること，フレームどうしはポインタで結合されており，状況変化に応じて適切なフレームが選択されること等しか主張しておらず，何故そのような表現形態がとられているかに関しての言及はないが，思考の効率化という観点から知識表現を上記のように見直すとまさにフレーム理論になるのである．つまり上の例は，Tweety, bird をフレームとし，これらのフレーム間に ISA リンクを張ったことに相当する．抽象状況内にある情報がフレームのスロットやその値に相当する．さらに，デフォールトの概念は状況依存の情報の省略とみなすことが可能である．

4.4 状況依存エージェントのモデル

以上のような考え方に基づき，状況依存性を最大限に利用した内部表現，推論機構，通信機構を持つエージェントモデルを作成することが筆者の目的である．状況依存性は知識表現，推論には都合がよいが，通信の場合は表現形態の異なるエージェントが通信することになるので配慮が必要である．例えば$\langle\langle$thirsty$\rangle\rangle$という内部表現を持つエージェントがそれを他のエージェントに伝える場合に単に

《thirsty》

でよいのか，

《thirsty, agent0011, location1, time16:00》

のようにしなければならないかは状況に依存する．あるいは，特定の個人を指すのに“あの人”，田村さん”，“部長”のどれを使うのが最も良いかも状況に依存する．つまり，通信の際には（送り手，受け手ともに）状況に関する推論が必要となる．

これはある意味でのオーバーヘッドではあるが，これまで全く通信のなかった新しいエージェント間の通信のモデルにはむしろ有効であると考えている．つまり，従来の単なるメッセージ交換にはない柔軟な通信が可能になると考えている．

5 まとめ

協調アーキテクチャプロジェクトの研究指針を示した．状況推論，オブジェクト，リフレクション，タイプ理論などからのアプローチを考えているが，それらのうち状況推論の考え方を示した．知識表現，推論，通信などを状況依存性の立場でとらえ直すことにより，環境に柔軟に対応できるエージェントの構成を目指している．ただし，まだ具体的な成果といえる段階ではない点は御了承願いたい．

謝辞

ここに述べられたアプローチの内，状況推論以外の部分は電総研 協調アーキテクチャ計画室諸氏のものである．ここに記すとともに，日頃の討論に感謝する．また，状況推論の考え方については千葉大学 土屋俊氏，Stanley Peters 氏を始めとする CSLI の研究者，ICOT STS WG メンバー諸氏との討論に負うところも大きい．

参考文献

[Bar89] Jon Barwise. *The Situation in Logic.* CSLI lecture notes, No. 17, the University of Cicago Press, 1989.

[CLNO90] P. R. Cohen, H. J. Levesque, J. H. T. Nunes, and S. L. Oviatt. Task-oriented dialogue as a consequence of joint activity. pages 203–208, 1990.

[Fau85] Gilles Fauconnier. *Mental Spaces.* MIT Press, 1985.

[FF86] Robert E. Filman and Daniel P. Friedman. *Coordinated Computing – Tools and techniques for distributed software.* （邦訳：協調型計算システム ー分散型ソフトウエアの技法と道具 立てー，マグロウヒル），1986.

[Gel85] David Gelernter. Generative communication in linda. *ACM Transaction on Programming Languages and Systems*, 7:80–112, 1985.

[GMP82] J. A. Goguen, J. Meseguer, and D. Plaisted. Programming with parameterized abstract objects in OBJ. In D. Ferrari, M. Bolognani, and J. Goguen, editors, *Theory and Practice of Software Technology*, pages 163–193. North-Holland, 1982.

[Gog87] Joseph A. Goguen. One, none, a hundred thousand specification languages. Technical report, CSLI, 1987.

[HS86] Takumi Hisano and Motoi Suwa. Synchronization and communication in the 'subject'. In *Logic Programming '85, Lecture Notes in Computer Science*, volume 221. Springer-Verlag, 1986.

[Lea79] Barbara H. Liskov and et. al. Clu reference manual. Technical Report TR-225, Lab. for Computer Sci., MIT, 1979.

[Min75] Marvin Minsky. A framework for representing knowledge. In Patric Winston, editor, *The Psychology of Computer Vision*. McGraw Hill, 1975.

[NOK91] H. Nakashima, I. Ohsawa, and Y. Kinoshita. Inference with mental situations. TR-91-7, ETL, 1991.

[NPS91] Hideyuki Nakashima, Stanley Peters, and Hinrich Schüzte. Communication and inference through situations. In *Proc. of IJCAI-91*, 1991.

[NS72] Allen Newell and Herbert A. Simon. *Human Problem Solving*. Prentice Hall Inc., 1972.

[NT91] Hideyuki Nakashima and Syun Tutiya. Inference *in* a situation *about* situations. *to appear in Situation Theory and its Applications, 2*, 1991.

[Ros87] Stanley J. Rosenschein. Formal theories of knowledge in ai and robotics. Report 87-84, CSLI, 1987. 現代思想 18:3 に翻訳.

[テ 75] ニコラス ティンバーゲン. 本能の研究. 三共出版, 1975.