

ネットセントリックCAD環境と共同設計

Net-centric CAD Environment and Collaborative Product Design

横地 夏樹

Natsuki YOKOCHI

株式会社ソフトウェアスaska (〒176-0023 東京都練馬区中村北三丁目15番16号
田丸ビル206号 E-mail:yokochi@swa.co.jp)

ABSTRACT. The purpose of this project is to provide a solution for collaborative design and engineering in the coming broad-band network age. Most of all current CAD systems are, so-called, stand-alone type, and are not prepared for the collaborative system functions by themselves. We have introduced a flexible client/server type of CAD system, where any member of a design/engineering team will be able to work together with the same component or product on a simultaneous /real-time basis. And then we propose the Net-centric CAD environment for the enterprise engineering activities in the near future.

1. 背景

(1) 現状CADの問題点

製品開発部門における、開発期間の短縮、設計品質の向上、開発コストの削減は永遠の課題であるが、CADはこの課題を解決する有力な手段として、今日では多くの製造業に導入されている。一方、CAD自体についても簡単な製図機能を提供するローエンドCAD(例えばVisioなど)から航空機産業や自動車産業に代表される大規模産業の高度な要求(ソリッドモデリング、デジタルモックアップなど)に答えるハイエンドCADまで数多くのCADが開発され、あるいは販売されている。

これらのCADの、道具としての機能、性能の追求は今やかなりのレベルにまで達しているように思われるが、一方で、仕組みとしてのCADはまだまだ幾つかの問題点を残している、あるいは内包しているように思われる。それらの問題の一つにCADによる設計者の「蛸壺」現象とよばれるものがある(図1)。本来設計作業は多くの設計者による共同作業によって、より高い生産性を得るものであるが、これが、CAD化によって各設計者は、自らの端末と作業ファイルにかかりきりになり、その間、その中で何が行われているか、設計リーダーもチームを組む他の設計者も十分に把握できない。小部品を一人の設計者が設計する場合も同様な状況で、CAD化後設計者と設計リーダー、設計管理者との間のタイムリーなコミュニケーションが減少しているといわれている。その結果、各設計者の結果を統合して検討する、より後ろの段階で多くの不具合が発生する。これがCAD化によって設計者個々の生産性は大幅に向上したかのように見えるのに、全体としての製品開発期間の短縮ではそれほど効果が現れていないといわれる要因の一つであろう。

このような問題は、CADシステムはワープロなどのアプリケーションと同じく、セッション中は本質的にスタンドアロン型であるということに起因している。セッションを離れた段階では、たとえばPDM(Product Data Management)やWFM(Work Flow Management)などのシステムと連携して、これらの問題を解決するために

様々な試みがなされている[1]。しかしCADシステム固有の問題として本格的に取り組み、成果を挙げている例は少ない。問題の所存は、CAD固有の仕組みの中にあるので、真の解決には、CADシステム自身が真正面からこの問題に立ち向かうことが必要であると考ええる。

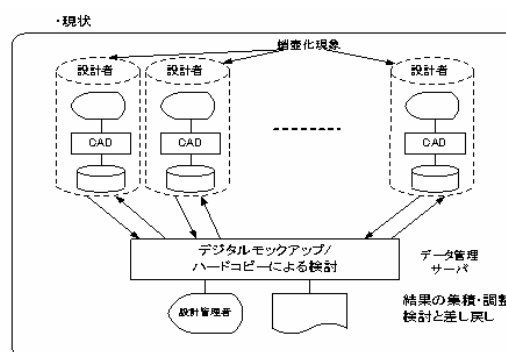


図1. 現状CADの問題点

2. 目的

(1) 共同設計CADの開発と実用化

更なる開発期間短縮とコスト削減のためにCADによる共同設計は有力な手段と考えられる。CADの普及は設計者が個別に設計作業を行う場面では、効果を挙げたが、前述のように製品開発の全期間、あるいは、開発部門全体の視点に立つと未だいくつかの課題を残している(図1)。

- ・個別の設計者が個別に設計作業に没入している段階ではリーダーまたは管理者は、個々の作業の状況を見ることが困難である。
- ・個々の設計結果を集めて、設計を完成させる段階で、膨大な調整作業が必要となることが多い。

などであり、これらは現行CADシステムの本質的な弱点となっている。これらを克服するために、CAD自身による「コラボレーション」の支援、すなわち共同設計CADの開発と実用化が急務である。本論分では、共同設計の実用化ために可能な方式を確立することを目指す。

中核的な機能としては、同じデータに対して複数の設

計者による同時共同設計機能である。伝統的な製図版による設計形態に喩えれば、超大型の製図版に複数の設計者がチームになり同時に取り付き、管理者と設計者が全体の設計状況を把握しながら、自らの担当作業を遂行しているイメージをシステム化することである。

(2) アプリケーションシステムにおけるシステムアーキテクチャの重要性の認識と確立

我が国におけるアプリケーションシステムの開発では、その外部機能を満たすことが重視され、システム機能や構造の一般化・抽象化は、ややもすれば軽視されがちである。言い換えると、アプリケーションシステムのアーキテクチャが深く追求された形跡がない。確固たるシステムアーキテクチャの議論がないままに、外部機能の追加と拡張を繰り返し、気がついたときには泥沼にはまりこんでいたアプリケーションシステムの事例は、枚挙にいとまがない。また、IT分野で主導的な役割を果たすためには、このシステムアーキテクチャレベルの技術力が不可欠であるが、世界市場で流通する我が国発のミドルウェアなどが少ないことの要因の一つがこのあたりに在るとすれば問題は深い。

本テーマへの対応についてもシステムアーキテクチャについての言及は不可欠であり、本論文ではこれを明確にする。また本テーマだけでなく、いわゆる「次世代CAD」を論ずるに当たっても、本システムで採用したアーキテクチャはひとつの提案になりうると考える。形状定義や属性定義などの作業的作業の領域の機能や性能の追求は早晩飽和し、次世代型CADへの転換がはかれるであろう。この次世代型CADでは、企業毎、製品毎により特化し、かつ設計ノウハウなどの知的作業領域に踏み込む機能が求められることになる。これらの機能が現行のCADシステムアーキテクチャ上ですべて可能であるとは思われない。今、次世代型CADのあり方にたいして、外部的な機能についてだけでなく、内部的なシステムアーキテクチャについての提案も必要であると考ええる。

3. 共同設計CADの方式

(1) リアルタイムなデータ共有の必要性

現行のCADシステムの殆どは、1部品（あるいは部位）- 1データファイリングユニット（以下DFU）- 1CAD設計担当者の対応で運用されている。セッション中はデータファイルの内容がコンピュータシステム上のメモリ作業域に展開されて、CADシステムはこれにアクセスする。前述のように、現行CADシステムは殆どがスタンドアローン型であり、他ユーザのメモリ作業域に直接アクセスすることは多くの場合不可能である。一方、CADによる共同設計の実現においては、個々のメモリ作業域の情報をチームメンバがリアルタイムに共有することが不可欠である。すなわち、他の設計者によるデータ更新が、直ちに自らのディスプレイに反映されること、そしてそれらのデータを参照したり、場合によっては更新することも可能であることが求められる。データを共有する仕組みとしては、次の二つの方式が考えられる。

(2) 個別メモリ作業域方式

この方式では、メンバが使用するCADごとに固有のメモリ作業域を有するが、その内容を常に同じ内容に保つことによって、メンバ間のデータの共有を実現する。すなわち論理的に一つのDFUを複数の設計者が個々の

物理的なメモリ作業域に展開してアクセスするが、その内容はメンバ間で常に同一であることが前提になっている。この方式の実装にあたっては、

- ・すべてのメンバが同じCAD処理コマンドを同じ順序で実行するようにして、各自のメモリ作業域の内容を同一に保つ。このために各メンバからの処理要求を一旦集中制御部が吸い上げて、それを各メンバに配送し、そこで実行されるようにする。
- ・メモリ作業域を管理するサブシステムが互いに連絡を取り合い、各メンバの個別メモリ作業域の内容が常に同一になるよう管理する。

などの方法が採られている。個別メモリ作業域方式では、個々のメンバのメモリ作業域の内容が同一であることを検証することが重要であるが、かなり難しい問題である。また途中からの共同設計への参加など、運用に絡む手続きも複雑なものがある。

この方式で共同設計を実現しようとする取り組みはすでに幾つかが報告されているが[1,2,3]、システムアーキテクチャは基本的にはスタンドアローン型である。

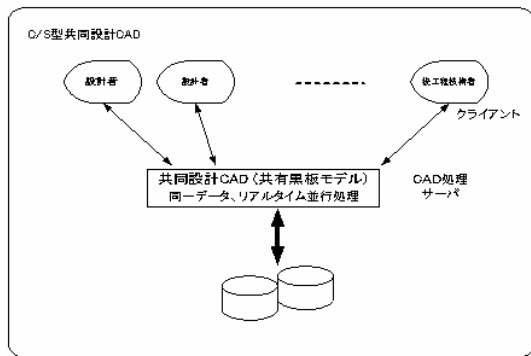
(3) 共有メモリ作業域方式（共有黑板モデル）

データ共有のもうひとつの方式は、ただ一つのメモリ作業域をチームメンバが共有する最も直截的な方法である。この方式では、メモリ作業域はサーバ上に存在して、チームメンバはクライアントからサーバに接続してデータにアクセスすることになる。この方式ではサーバでのデータアクセスの排他制御が重要な問題となるが、クライアントはいつでもサーバの最新状況にアクセスできるので、任意時点での共同設計への参加、退出が可能になるなど運用面でのメリットは大きい。

現状のCADシステムでは、1単元のCAD処理（形状の計算処理など）でも、かなり広範囲のメモリ作業域にアクセスするので、CAD処理機能もサーバ上に実装することが望ましい。したがって、この方式のCADシステムは、対話と表示機能を中心とするクライアントシステムとCAD処理と共有メモリ作業域管理を中心とするサーバシステムより構成される。このようにシステム基本構成はクライアント/サーバ方式であるが(図2)、従来のCADに対して、操作ヘッド(ディスプレイヘッド+キーボード、マウスなどの入力装置)を複数化した「マルチヘッドCAD」として眺めることもできる。

このようなクライアント/サーバ型CADシステムが可能になるためには、ハードウェアの性能向上、コストダウンに負うところが大きい。2GHzを超えるCPU速度、GB以上のメモリ、2ないし4CPUのSMP(Symmetric Multi-Processor)がリーズナブルなコストで利用できるようになった。またLAN環境では、1000baseのEtherが実用期を迎えている。数人から十数人のチームメンバからなる共同設計をまかなうには十分な性能である。

本論文で述べる共同設計CADはこの「共有メモリ作業域」方式によるものである。

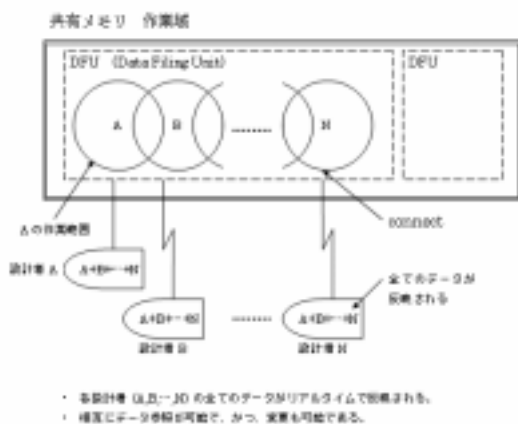


4. 共有メモリ作業域における二つの共同設計モデル

本システムでは、次の二つの共同設計モデルをサポートする。いずれのモデルでも「蛸壺」現象の解消に有効である。

(1) 単一DFUモデル

一つの部品/部位を分割することなく一つのDFUとして複数の設計者が同時に設計するモデルである(図3)。大規模の設計対象を分割することなく複数の設計者が同時並行に設計作業を進める、いわゆる「水平協業」を支援するものである。設計リーダー、設計管理者もメンバーの一員としてスーパーバイズしたり、アドバイスしたりすることができる。



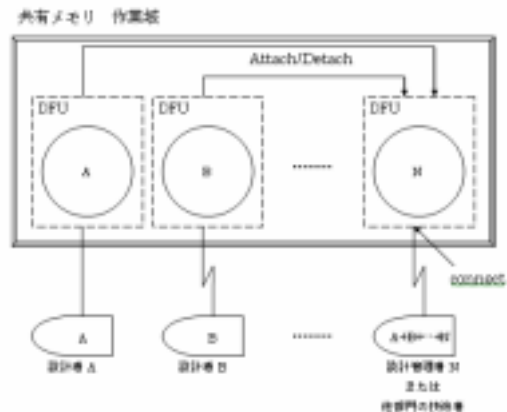
(2) 複数DFUモデル

サーバの共有メモリ作業域に各設計者が受け持つDFUを保持し、個別にそれに対して設計作業を行うが、必要に応じて他の設計者のDFUをアタッチして、その設計者の設計状況をリアルタイムに参照できる(図4)。アタッチされたDFUは更新することは不可能として、個々の設計者にとってはスタンドアローン型と同様の安全性を確保する。同時に、参照に対しては、単一DFUモデルと同様のリアルタイム性が満たされ、かつ随時必要な部位(DFU)のみを選択して反映する自由度が得られる。従ってスタンドアローン型からの移行として設計者に受け入れられやすく、また次に述べるように、「垂直協業」にも適用できる。

図 4 において設計者 A、B を開発部内の技術者に、設計

管理者 N を後工程の技術者とみなせば、工程間の協調設計が実現する。生産準備部内の技術者 N は、設計者 A、B が定義した形状より、タイムラグなしに型設計や治具設計など、生産準備作業に着手できる。

このように、前工程の設計活動を常にリアルタイムで注視することで、生産準備や生産技術的な検討を同時に進めることができる。これは、垂直協業の形態に他ならず、部門間/工程間コラボレーションが実現できる。なお、システムが許容する共同設計者数の最大値の範囲内であれば、技術者 N に相当する者は複数可能である。



5 . SWA-AF (SoftWorks Aska Application Framework)

上に述べたような概念と機能を持つ「共同設計 CAD」はソフトウェアスアスカ社の製品である SWA-AF 上で開発された。以下に SWA-AF の概要を述べる。

(1) SWA-AF の目的

SWA-AFは、CAD/CAMなどで代表される対話型グラフィックアプリケーションの開発ならびに実行のためのフレームワークである。アプリケーションシステムそのものではなく、アプリケーション開発を強力に支援するフレームワークであり、同時に、アプリケーションの実行環境においては、システム全体の高い効率とシステムの変更・拡張に迅速に対応できる柔軟性を実現するためのミドルウェアである。

高いコスト・パフォーマンス、将来の拡張性、あるいは、他システムとの接続性やシステムの環境変化に迅速に対応し得る柔軟性など、極めて高度なシステム要件が求められているアプリケーションにおいては、十分かつ精微な「システム設計」が必要とされる。市場には、優れた機能を持った「コンポーネント」が多く存在するが、これらを寄せ集めただけでは、高度なシステム要件には応えられない。現実には、多くの優れたコンポーネントを集めながらも「システム設計」が軽視されたため、苦渋をのんだ、あるいは、のんでいるアプリケーション開発の事例は多く見受けられる。

SWA-AFは、対話型グラフィックアプリケーションを開発するために必要な機能を提供するだけでなく、「システム設計」のための手法（システムアーキテクチャ）を提供するものである。高度なシステム要件を必要とする、いわゆるヘビーデューティなシステムの開発・運用において、これらの基本的問題について強力な解決手段を提供するものである。

(2) SWA-AFの開発にあたっての指針

SWA-AFの開発にあたっては、外部要件的には、いわゆる「次世代CAD」への対応、内部的には、採り得る情報処理技術として、分散処理を基本にしたネットワーク上のシステム構築技法が中心的なテーマになった。結果的には、これらの二つはほどよく調和した。

a) 「次世代CAD」への対応

SWA-AFのシステムアーキテクチャの策定は、いわゆる「次世代CAD」のシステムアーキテクチャの検討が契機になっている。

設計・開発の生産性をさらに高めたいという理由から、企業ごとあるいは部門ごとに、さらには開発する製品の種別により、カスタマイズされた専用CAD機能を使いたいと言う要求、言い換えれば、企業あるいは、設計者が持つ固有の設計ノウハウをCADへ組み込みたいと言う強い要請がある。また、同じ理由から、リアルタイムでの共同設計が可能な環境や協調開発（コラボレーション）環境を整備・実現したいという根強い要請がある。

これらの要請に対応可能なCADシステムを「次世代CAD」と呼ぶならば、これを現行のモノリシックなシステムアーキテクチャ上に実現することは、困難であると考えた。「次世代CAD」においては、これらの要請、課題に対応するため新しいアーキテクチャが策定されねばならず、SWA-AFはこれに対する回答のひとつである。

図5は現行CAD、次期CADから次世代CADへの発展過程の一例を示す。

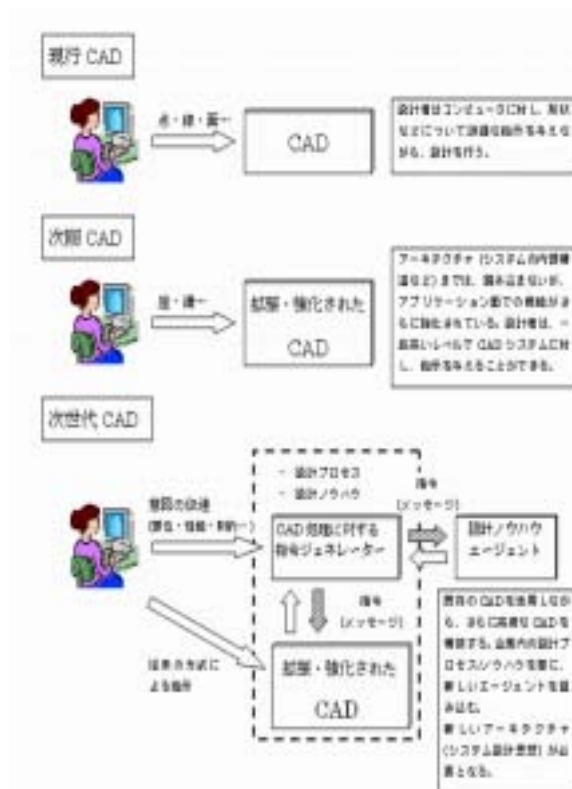


図5. 現行CADから次世代CADへ

SWA-AFでは図における「次世代CAD」のイメージに柔軟に対応することを一つの命題とした。

b) エージェント指向

ネットワークの普及により、情報システムの有様が変化した。ネットワーク上のノードに必要な機能を分散し一定のプロトコルでそれらを接続して協調動作させる方式で情報システムを構築する手法は大成功をおさめた。

代表的な例はクライアント/サーバ型分散ネットワークシステムである。

ここで用いられる概念や技法は、一つのノード（一つのコンピュータ）上で動くアプリケーションシステムの構築にも取り込むことができる。ネットワーク上の各ノードに分散したサブシステムに対応して、幾つかの「プロセス」を作成し、あるプロトコルでそれらの間のコミュニケーションをとりながら協調動作させることで、アプリケーションを可能ならしめる。ネットワーク上のシステムと同様に、アプリケーションシステム内にクライアント型の「プロセス」やサーバ型の「プロセス」が混在し、かつ、それらの組み合わせは（動的組み合わせを含めて）柔軟である。手続き型の集成であるモノリシックなシステム構造に対して、機能拡張や、環境変化への対応などにおいて優位な位置にある。それは分散型ネットワークシステムと同様である。

SWA-AFでは将来の発展をこめて、これらの「プロセス」を「エージェント」と呼ぶことにしている。また、「プロセス」間のコミュニケーションには、SWA-AF独自の「メッセージ」で行う。このメッセージはあらかじめ形式は定められているが、新たな意味をもつメッセージを任意に設定できる自由度をもつ[4]。

(3) SWA-AFのシステムアーキテクチャ

図6はSWA-AFのシステムアーキテクチャの概念図である。

SWA-AFの主要構成要素は「エージェント」と呼ぶメッセージ駆動型実行体である。エージェントは、「メッセージ交換・解釈機構」を通じて自分宛のメッセージが配布されると、それを受け取り、内容を解釈して実行する。実行が終了すると、また次のメッセージの到来を待ち受ける。SWA-AFでは、このようなエージェントの集合が、相互にメッセージを交換し、協調動作して必要な処理を実行する。

これをオペレーティングシステムの側からいえば、OSのもとでいくつかのプロセスがメッセージ交換と処理を通じて、互いに連携し、ひとつのまとまったシステムを構成するメッセージ駆動のマルチプロセスシステムと見ることができる。これらのプロセスをSWA-AFでは「エージェント」と呼ぶ。

どのようなエージェントが必要かは、アプリケーションに依る。例えば、形状処理を目的とするCADでは、対話エージェントと形状処理エージェントがあればよい。対話エージェントは対話操作によって得られた入力データをメッセージの形に編集し、形状処理エージェントに引き渡す。形状処理エージェントは、そのメッセージを解釈して、必要な形状処理を行い、結果を「データ共有管理機構」を通じて共有メモリ上の作業域に格納する。さらに形状処理エージェントはグラフィックディスプレイ上に結果を表示するためのメッセージを対話エージェント宛に発する。対話エージェントは、それを受け取り、共有メモリ上の最新データより画面の表示を行う。

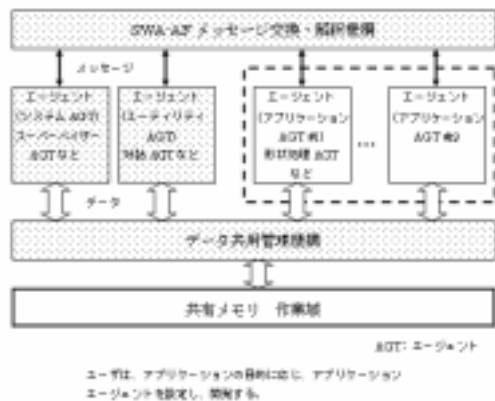


図 6 . SWA-AF システムアーキテクチャ概念図

（４）SWA-AF が提供する機構とエージェント

SWA-AF では、特定のアプリケーションを担うエージェントを除いて、システムを構成する基本的な機構とエージェントは全て提供される。（図 6 の網掛け部分）

「メッセージ交換・解釈機構」では、OS からの独立性と性能を高めるために、独自の方法を採用している。メッセージは人による理解を容易にするため、テキスト文字で記述することを基本としている。メッセージは一定の書式を持つが、これはエージェントの開発者に対し、メッセージ理解の負荷を最小にし、メッセージ交換・解釈機構によるインタープリテーションの高速性を図るためである。

メッセージの集合をメッセージマクロとして扱えば、手続型スクリプト言語として機能する。

メッセージによるエージェント間の情報交換は、大量のデータ交換に不向きである。大量データの高速処理を可能にするため、「データ共用管理機構」が整備されている。ここでは、単なるメモリ管理機能以外に、データ構造の表現機能も用意されているので、多様なアプリケーションに対応できる。

システムの制御や管理のために必要となるシステム基本機能は、エージェントとして提供されている。エージェントは、

- ア) システムエージェント
- イ) ユーティリティエージェント
- ウ) アプリケーションエージェント

に三分類されるが、ア) とイ) は予め用意されている。

システムエージェントには、

- ・スーパーバイザーエージェント
- ・コンソールエージェント
- ・アドミニストレーターエージェント
- ・ネットワークメッセージエージェント(送信・受信の 2 個のエージェント)

がある。

ユーティリティエージェントとしては、対話エージェントが用意されている。対話機能は本来アプリケーションに依るが、この対話エージェントでは、豊富なメッセージ機能により、メッセージマクロの記述だけで多くのアプリケーションの GUI 要件に対応できるようになっている。

（５）SWA-AF でのアプリケーションシステムの設計

SWA-AF のもとでは、エージェントはメッセージの交換を通してのみ、相互作用が行われる。いったん渡されたメッセージは、他エージェントの干渉を受けることなく、

独立性と一定の権限をもって処理を行う。また、複数のエージェントを一連のメッセージで連鎖してひとつのまとめた仕事を完結することもできる。このシステムの挙動は実世界における業務プロセスときわめて酷似しており、この特性を生かして、明確なアプリケーションの設計を行うことができる。エージェントはその下位にサブ・エージェントを持つことができる。サブ・エージェントの動作原理はエージェントと同じである。

一般的な、要件定義に続いて

- ・この要件を満たすいくつかの部門がフラットに並ぶ仮想組織を設定する。
- ・各部門の機能（ファンクション）を定義する。
- ・部門間にまたがる業務プロセスを定義する。
- ・業務処理にあたって必要な標準ルールなどを設定する。
- ・組織としての挙動をシミュレートとして、妥当性・最適性を評価する。

上の結果を受け、コンピュータ上のシステムとして、焼きなおす作業を行う。すなわち、具体的なシステム設計の実行である。

- ・部門をアプリケーションエージェントに対応させ、その機能を定義する。エージェントの機能定義において、さらに下位の組織を想定したほうがより機能が明確になるのであれば、その下位組織をサブ・エージェントとして機能定義する（サブ・エージェントの動作はエージェントと同じ原理）。
- ・各エージェント/サブ・エージェントについて、その機能を果たすため、自ら処理すべきメッセージを定義する。
- ・メッセージのやりとりとそのメッセージの意味を通して、システム挙動の机上シミュレーションを行い、妥当性・最適性の評価を行う。これらの作業を経て、エージェント/サブ・エージェントの概念レベルの機能が確定する。言い換えれば、処理すべきメッセージの集合と個々のメッセージの意味が確定する。

エージェントの機能定義が完了すると、次は実装を考慮したシステム設計のフェーズである。アプリケーションシステム全体として必要な共有メモリ、外部ファイルを用いる場合の標準ルール、状況テーブルなどの設計を行う。

SWA-AF では、配下のアプリケーションエージェント間の相互作用はメッセージ中心で行われるので、メッセージの定義が確立した後は、エージェント毎に独立して、システム設計と開発が行える。また、SWA-AF は、このようにシステム設計でもひとつの枠組みを与えており、システムの概念設計段階から、システムの仕組み、挙動のイメージを具体化し、早期に机上シミュレーションを可能にする。このことから、システム設計上の欠陥や不具合を早期に発見できる。開発期間の短縮のみならず、稼働後の変更・メンテナンスあるいは拡張を容易に行うことを可能とする。

エージェントを駆動するメッセージはテキスト文字であるため、メッセージはネットワークに対し高い透過性を持つ。ネットワークの他ノードにあるアプリケーションエージェントに対し、メッセージを送ることで、そのサービスを受けることが可能である。

（６）OS からの独立性

SWA-AF は、Windows と Linux、どちらの OS のもとでも稼動する。これら OS の差異を SWA-AF が吸収することに

より、SWA-AF 上に構築されるユーザのアプリケーション (アプリケーションエージェント) は、これら OS から独立していることを意味し、ユーザはその資産である一種のアプリケーションエージェント群を開発・管理すればよい。

同時に、ユーザのシステム選択の幅を広げ、また、将来のシステム稼働環境の変化にも柔軟な対応性をもたらす。

6 . SWA-AF の共同設計への適用

(1) サーバ

SWA-AF で C A D システムを構築する場合、必要なエージェントは、形状処理を主な仕事とする C A D 処理エージェントと、メニュー選択、入力、結果の表示を主な仕事とする対話エージェントである。一つの SWA-AF のもとで、複数の対話エージェントを動かすことも、SWA-AF の設計思想上制約を受けるものではない。ただし、一般に一つのコンピュータには、キーボードやマウスなどの入 (出) 力装置は一組しかなく、実際に複数の設計者が同時に操作することは不可能である。そこで、X Window System におけるリモート X サーバのように、グラフィックディスプレイを含む入出力部を、ネットワークを介して、外部のノード (クライアント側) に設定できれば、複数の設計者が同時に操作できる「共同設計 C A D」が実現できる。

リモート X サーバの場合は、いわゆるシン・クライアント (Thin Client) であるが、このシステムではできるだけサーバの負担を軽くするためと、Linux-Windows などの異 OS 間接続を可能にするために、クライアント側にも多くの処理を委譲している。そのためローカルで対話の窓口となる「マスタ対話エージェント」からクライアントでの処理部分を取り除いたような「クライアント I / F エージェント」がサーバ上で動作する。

SWA-AF は、もともとマルチプロセスで動作するので、メモリ作業域は複数のエージェントで共有使用されることを前提にしている (データ共用管理機構)。そのために排他制御などの仕組みはすでに実装されているが、単一 D F U モデルや複数 D F U モデルをより効率よく扱うための機能拡張がなされている。

複数 D F U を共有メモリ作業域に取り込む場合は、C A D 処理エージェントを複数個起動し、サーバでの処理効率を上げることができる。特にマルチプロセッサ搭載のサーバの場合、担当する D F U ごとに物理的な並行処理が可能となり効果が大きい。

(2) クライアント

クライアント側では、対話のためにサーバ側の「クライアント I / F エージェント」と対をなすモジュールが動作すればよいが、この「共同設計 C A D」では、クライアント側でも SWA-AF を動かして、このモジュールを SWA-AF のエージェントとして組み込んでいる (「クライアント対話エージェント」)。この「クライアント対話エージェント」は、前述のようにサーバの負荷を軽減するためと、異 OS 間接続 (たとえば Linux サーバと Windows クライアント) を可能にするために、一般のグラフィックライブラリ以上の処理機能を果たしている。

図 7 に共同設計 C A D の構成図をしめす。この図のクライアントにおいて、C A D 処理エージェントとデータ共用管理機構以下は必ずしも必要としない。

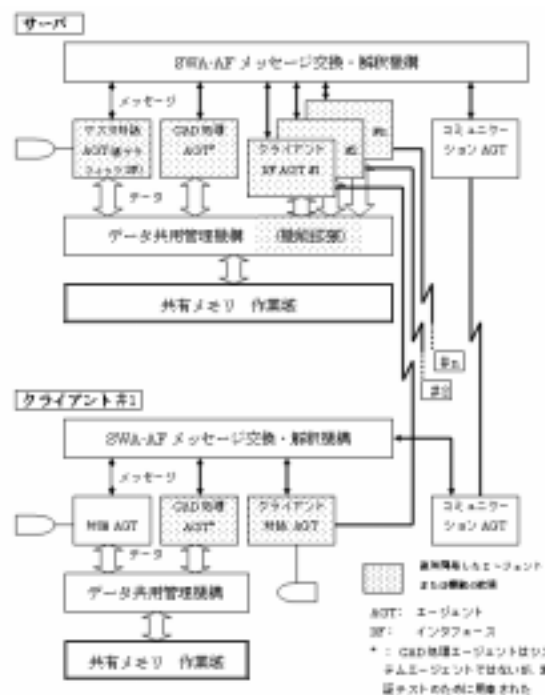


図 7 . 共同設計 C A D のシステム構成

(3) 開発した共同設計支援機能

上述のシステム構造のもとに、共同設計を支援するために次のような外部機能が開発された。

a) 共同設計 IN/OUT 機能

任意のサーバについて、共同設計に参加・退出する機能である。

b) 共同設計支援表示制御機能

形状を担当した設計者毎に色を替えた表示、あるいは、一時的にほかの設計者の作業結果を自らのディスプレイに反映することを回避するなど、作業性を向上するための表示制御機能である。

c) 共同設計支援データ制御機能

自ら作成したデータを他の設計者が変更することを不可とし、あるいは、それを解除する。また、共同設計している他の設計者には関係のない自分の検討のためのデータ生成機能も含まれる。

d) コンファレンス機能

それぞれの操作ディスプレイに同一の表示を行い、共同設計参加者が設計検討を行う場合に用いる機能である。

・コンファレンス IN/OUT 機能

コンファレンスへの参加・退出機能

・オーナーシップ機能

オーナーシップを持つ設計者が画面表示を変更 (拡大・回転など) すると、即時にコンファレンスに参加する設計者の画面も同一表示がなされる。オーナーは常にチーム内一名であるが、メンバであれば誰でもオーナーシップを取ることができ、自由に交代することができる。

・マーキング機能/ハイライティング機能

オーナーシップを持つ設計者だけが使える機能でコンファレンス参加者全員の画面に反映される。マーキング機能は、オーナーが選択した要素の色が変わる。ハイライティング機能はオーナーのマウスがヒットしている時間だけヒットされた要素の色が変わる。

e) コミュニケーション機能

共同設計に参加している設計者間で操作コンソールを用いたコミュニケーション機能で二つの機能を提供する。

- ・チャット機能
二人の設計者間でテキストによる会話を可能にする。
- ・共用掲示板機能
設計者が共用掲示板に書き込むことにより、情報伝達する機能である。

(4) 「共同設計CAD」の稼働環境

本システムは次のハードウェア、オペレーティングシステムのもとで稼働する。

- ・ハードウェア
 - IBM PC/AT 互換機
 - CPU Intel Pentium Processor 466MHz 以上
または互換相当品
(サーバでは1GHz 以上が望ましい)
 - メモリ 128MB 以上
(サーバでは256MB 以上が望ましい)
 - ディスプレイ / カラー XGA / 65000 色以上
 - ・オペレーティングシステム
 - Linux カーネル 2.4 以上
glibc 2.2.4
XFree86 4.0 以上
 - Windows 2000/XP
- サーバ、クライアントが同一 OS である必要はない。

(5) 実証テストと評価

開発された「共同設計CAD」について、実務へ適用できるレベルの基本性能（レスポンスなど）と必要な基本機能を達成しているかどうかを検証した。

a) 共同面張り作業

- ・モデル
曲線数 13,000 本以上、曲面数 850 面以上の実の自動車内板部品
(自動車内板部品としては規模の大きい部類に属する)
- ・テスト環境
 - サーバ
Intel Xeon 2GHz × 2 マルチプロセッサ
メモリ 512MB
 - クライアント
Intel Pentium 266MHz ~ 1GHz の CPU、
160MB ~ 256MB までのメモリを持つ4台のデスクトップおよびノートパソコン
 - LAN
100BaseT
- ・検証機能
 - 面張り
すでに十分大きなデータサイズになっている共有メモリ作業域上の DFU に対して4クライアントが同時に新たなNURBS 曲面を定義して生成する。
 - 結果：従来のCAD (1台のCAD が単独で動作している) に対して遜色のない応答が得られた。

コンファレンス

四面図表示 (一画面上で同一モデルを4回表

示したのと同じ) で5台のコンピュータに同一画面を同時に配送するコンファレンスでレスポンスを検証する。

結果：従来のCAD に対して遜色のない応答が得られた。

b) 自動車フロントピラー部とフードインナー部の個別設計時のモニタリング(複数 DFU 共同設計モデル)

- ・モデル
 - DFU 1: 自動車フロントピラー部内板データ
(クライアント対話 AGT で作業)
 - DFU 2: 自動車フードインナー部内板データ
(クライアント対話 AGT で作業)
 - DFU 3: DFU1 と DFU2 をアタッチ (マスタ対話 AGT でウォッチ)
- ・テスト環境
 - サーバ
Intel Pentium 1GHz M プロセッサ
メモリ 256MB ノートパソコン
 - クライアント1
Intel Pentium 800MHz
メモリ 256MB ノートパソコン
 - クライアント2
Intel Pentium 266MHz
メモリ 160MB ノートパソコン
 - LAN
100BaseT
- ・検証機能
 - 総合的な機能の確認
 - 結果: アタッチ/デタッチなど必要な基本機能は計画通り作動した。

c) 評価

今回の開発と検証にあたり、クライアント/サーバ方式と黒板共有方式をインフラにした共同設計CADシステムは十分実用に耐えると思われる。今後CPU 速度はさらに向上し、マルチプロセッサパソコンの普及も進展すると考えられ、またLAN も1000Base が実用期に入りつつあるので、この方式の共同設計CADの実用化のための環境は整ったと思われる。

さらにこの方式では、データの集中管理などセキュリティ上のメリットなども考えられるので、現状のCAD 導入による幾つかの問題点の解決に寄与できる。

今後の課題はCAD として必要な機能の充実をはかることである。CAD 処理に必要な強力な機能を持つアプリケーションを豊富に提供することである。

7. おわりに：ネットセントリックCADとしての展開

図7の共同設計CADのシステム構成で、サーバ側でクライアント対話エージェントを、クライアント側でクライアントI/Fエージェントを起動すれば、サーバ、クライアントともまったく対称な形態になる。すなわちどの設計者端末も同時にクライアントにもサーバにも成りうるという構成が実現できる。このような形態をクライアント/サーバCADの先にあるものとして、「ネットセントリックCAD」と呼ぶことにする。

SWA-AFのもとで開発した「共同設計CAD」では、スタンドアローン形態から、クライアント/サーバ形態、さらにネットセントリック形態までユーザが動的に選択・構成できる。

クライアント/サーバ形態では、設計者は常にある特

定サーバ(=設計チーム)のもとで仕事をするようになる。このことはセキュリティ管理など運用面で、様々な利点をもたらすが、設計者は、ある意味で閉ざされた共同設計環境の中にいることになる。

これに対し、ネットセントリックCAD形態では、特定のサーバに縛られて仕事をするという意識はなく、必要で可能なら、どのサーバのもとでも共同設計に加われるという開かれた環境が提供される。

運用での困難な面もあるが、これからのブロードバンドネットワーク時代にむけての設計環境として十分吟味の対象になり得ると考える。また特別のサーバを導入する必要がないという面からも、現行のスタンドアローン形態からスムーズな移行ができる。

CAD技術そのものは、すでに成熟したものとみなされ、もっぱら質の向上とか周辺システムとの連携などに力が注がれているように見られるが、CAD自身で業務の変革をもたらすような発展はまだまだありうる。

8. 参加企業

参加企業：株式会社ソフトウェアスaska

契約名：エージェント協調型CADフレームワークの共同設計への応用

9. 参考文献

- [1] 綾日天彦：インターネット時代の協創設計，p98，共立出版，1996
- [2] 川添浩史：CADとチーム設計，日経メカニカル，1995年3月6日号，p78(1995)
- [3] 川添浩史：CADとチーム設計，日経メカニカル，1995年3月21日号，p88(1995)
- [4] (株)ソフトウェアスaska，SWA-AF システムガイド，1998