N92-16604

Design Concepts for Integrating the IMKA Technology with CLIPS

David Scarola

Carnegie Group, Inc

Abstract. This presentation will share our experiences in evaluating the technical alternatives for integrating the IMKA frame-based knowledge representation system with CLIPS. The Initiative for Managing Knowledge Assets (IMKA), consisting of Carnegie Group, Inc., Digital Equipment Corporation, Ford Motor Company, Texas Instruments Incorporated and U S WEST, Inc., was formed to foster the cooperative funding and development of a software technology that will meet each company's and their clients' needs for capturing and managing complex, corporate-wide knowledge. The IMKA Technology is a frame-based knowledge representation system for developing knowledge-based applications.

We've explored various models for integrating the IMKA technology with CLIPS. Integrating the IMKA technology with CLIPS allows application knowledge to be encoded naturally using both frames and rules, and allows the knowledge stored in frames to be reasoned about using rules. Integrating a frame-based system with a RETE-based rule system is a challenging task because the approach to accessing data is very different in each system. We found three integration models that can be used to address the different data access methods of IMKA and CLIPS. This presentation provides an overview of the features of the IMKA technology, describes the challenges of integrating the IMKA technology with CLIPS, and discusses the three integration models and the circumstances under which each is appropriate.

NOTE: I apologize for not having a paper prepared for the conference proceedings. I just ran out of time. However, my slides are included and they contain a fair amount of detail. If anyone would like more information on either the IMKA Technology or the integration models discussed in the slides, feel free to contact me at either **scarola@cgi.com** or **(412) 642-6900.**

Overview

1. History and background on IMKA.
2. The IMKA Technology.
3. The design goals for integrating the IMKA Technology and CLIPS.
4. The challenges involved in the integration.
5. The models that we've used.
6. The circumstances under which each model is appropriate.
7. Open issues and technical challenges.

History and Background

- IMKA: Initiative for Managing Knowledge Assests
- Participants:
  - U S WEST, Inc.
  - Digital Equipment Corporation
  - Ford Motor Company
  - Texas Instruments
  - Carnegie Group, Inc.
- Purpose: Foster the cooperative funding and development of a software technology to meet each company's needs for capturing and managing complex, corporate-wide knowledge.
- The IMKA Technology: A frame-based knowledge representation system.
- Industry participation: Includes Quintus Corporation and AICorp.
- IEEE Working Group P1252. For more information, contact Mitch Smith at (303) 541-6133

The IMKA Technology

The feature set:

- Basic entity is a frame.
- Frames contain attributes, relations, and messages.
- Class and instance frames.
- Class/subclass hierarchy.
- High degree of dynamicity in the frame base.
- Users can define their own relations.
- Slot and value inheritance.
- Users can define customized inheritance.
- Context mechanism.
- Demon mechanism.
- Meta-Knowledge.
- Path following.
- Persistent frame storage mechanism.
- RDB interface.
- Designed to be used with C++ & C.
- User-defined C++ types.

Design Goals

- IMKA allows powerful reasoning systems to be developed.
- IMKA plans to integrate with multiple rule languages.
- Integrating IMKA and CLIPS would provide a powerful hybrid system that contains both frames and rules.
    - Enhances CLIPS by allowing rules to match against frames.
    - Enhances IMKA by providing it with rule-based reasoning.
    - Allows the appropriate representation of knowledge.
    - Maintains the uniqueness and power of each system.
    - Supports the external data storage mechanisms that IMKA provides.
    - Is highly-integratable and portable.

Frame/Rete Data Access Models

Frame Data Access Model

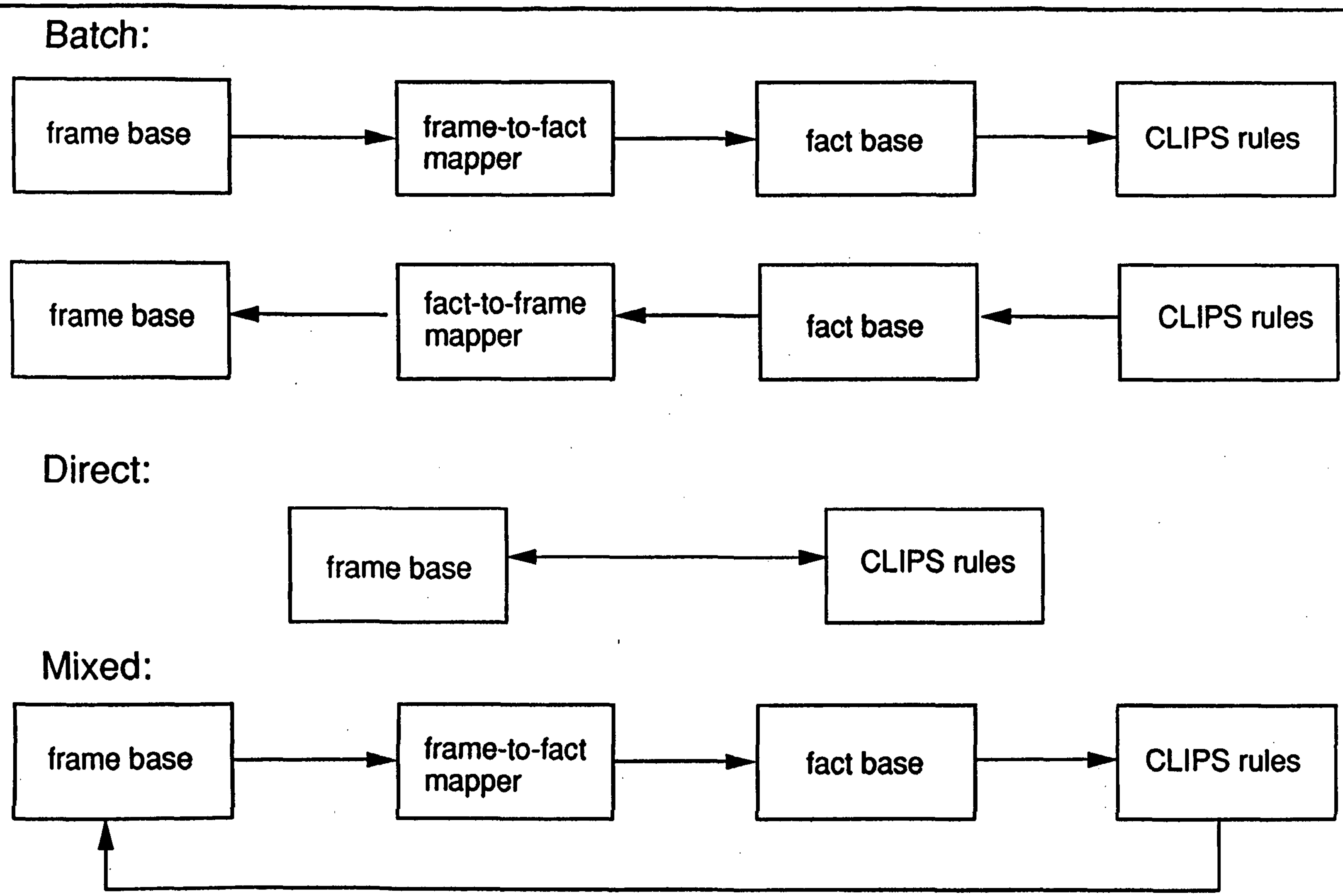
- Setting a value in a frame is basically a passive operation.
- When a frame value is read, dynamic frame mechanisms are invoked to return the correct value.
- For subsequent reads, these mechanisms are re-invoked to return the correct value.
- Due to the high degree of dynamicity in the frame system.

CLIPS (Rete) Data Access Model

- Reading a value from an object is a passive operation.
- When a value is written to an object, the modified object is propagated down the RETE network.
- The propagation requires that the correct values be available.
- Due to the caching mechanism for fast matches that RETE requires.

- Challenge: Matching rules against frames requires that the dynamic frame system features be invoked at write time so that the correct values can be matched by RETE.

Three Models of Integration



Batch Model

- Approach:
  - Build an IMKA frame base.
  - Build CLIPS rules to match against facts.
  - Run frame base until it is ready for rule-based reasoning.
  - Execute a function which maps the frame base into the fact base.
  - Run the rules to match against the resulting fact base.
  - Modify facts on the RHS of the rules.
  - Execute a function which maps fact base into frame base.
- Matching dynamic frame features in RETE: The dynamic features of the frame system are invoked at map time and cached in the fact base.
- Limitations: None -- allows all dynamic frame features to be matched by CLIPS rules

Direct Model

- Differences from BATCH model:
  - No facts -- frames are matched directly by CLIPS.
  - Interface is on-going instead of a one-time mapping.
- Approach:
  - Build IMKA frame base.
  - Build CLIPS rules to match frames using Deftemplate syntax.
  - RETE network matches frames directly instead of matching facts.
  - When frames are updated to contain new value, RETE must be informed of which frames are updated and need to be re-matched.
- Matching dynamic frame features in RETE: An interface must be supported by which the frame system informs the RETE network which frames need to be repropagated.
- Limitations: Certain dynamic frame features cause other frames to be updated which can't always be identified.

Mixed Model

- A combination of BATCH & DIRECT:
  - Like BATCH, CLIPS matches facts instead of matching frames.
  - Like DIRECT, the interface is on-going instead of a one-time map.
- Approach:
  - Build and IMKA frame base.
  - Build CLIPS rules to match against facts.
  - Each frame is assigned a corresponding fact (internally). Whenever the user modifies a frame, the corresponding fact is also updated.
  - User invokes CLIPS to match the facts which correspond to frames.
  - Modify frames on the RHS of the rules.
- Matching dynamic frame features in RETE: When a frame is modified, its corresponding fact and all other facts which correspond to frames which are effected by that modification must be updated.
- Limitations: Certain dynamic frame features cause other frames to be updated which can't always be identified.

Appropriateness of Each Model

Batch:

- Easiest model to Implement.
- No modifications to the internals of either system.
- Rules are run a single time.
- All dynamic frame mechanisms available for matching.

Direct:

- Hardest model to implement
- Modification to rule language and frame language internals.
- Memory is a premium.

Mixed:

- Modification to frame system internals.
- Rules are run multiple times.

Open Issues and Technical Challenges

- Dynamic IMKA features which are difficult to integrate with CLIPS:
  - User-defined inheritance
  - Demons
  - Contexts
  - Dynamic classes and slots
- Limitations in the CLIPS Deftemplate external interface.
- C++ name mangling and overload types.
- CLIPS object system 5.0 is not integrated with rules.
- Backward chaining.
- Common syntax between frames and rules.
- Non-RETE inferencing system.

For additional information contact Dave Scarola at
scarola@cgi.com or call (412) 642-6900.

406