Decentralized network management using distributed artificial intelligence

Eduardo Ferreira Mocelim (09238071)
Fábio César Ariati (10101182)
Felipe Duarte Silveira (10103190)
Fernando Almeida (10101184)

Federal University of Santa Catarina

Decentralized network management using distributed artificial intelligence

(Based on the reference – Decentralized network management using distributed artificial intelligence. FL Koch, CB Westphall. Journal of Network and Systems Management 9 (4), 375-388, 2001.)

Outline

1. INTRODUCTION
2. DESIRED FEATURES
3. INTELLIGENT AUTONOMOUS AGENTS
4. NETWORK MANAGEMENT AND AUTONOMOUS AGENTS
5. IMPLEMENTATION
6. CONCLUSIONS

1. INTRODUCTION

This presentation describes the research about how to delegate as much work as possible to the machine, using Network Administrators as knowledge engineers who teach the machine how it should perform its work. The approach presented here is based on Intelligent Autonomous Agents.

2. DESIRED FEATURES

- Objective: Develop a decentralized computer network management platform with concepts from distributed artificial intelligence
- Guidelines: High degree of adaptability, mobility, module reusability, self-generation and environmental plasticity.

2. DESIRED FEATURES

- From the guidelines, were inferred seven "commandments"
  - Agents must be truly "autonomous": any agents must be able to control itself and its own actions
  - Agents should be "goal-driven" and "rule based": A set of rules constitutes the knowledge base for any administration, and "goals" are the motivations that the agents have to work with.

2. DESIRED FEATURES

- From the guidelines, were inferred seven "commandments"
  - The Agent environment must behave as a "global knowledge base": each data item or item of information stored by one agent is "sharable" throughout the whole system (with other agents).
  - Agents should be capable of learning new skills: From other agents, the community of agents, the global community or human beings. These new rules should be dynamically stored in Agent's local knowledge databases.

2. DESIRED FEATURES

- From the guidelines, were inferred seven "commandments"
  - Agents should be self-generating: They should adapt themselves to new environments and generate sub-sets of their own characteristics creating new elements for new problems. The goal is to reduce human interaction as much as possible.

2. DESIRED FEATURES

- From the guidelines, were inferred seven "commandments"
  - They should interface with commercial solutions using standard protocols: This means that Agents should know how to interact with well known protocols such as SNMP and CMIP and how to interact with other existing management systems.
  - Agents should have a human-friendly interface: The friendly interface between humans and machines is a natural-language-like protocol.

3. INTELLIGENT AUTONOMOUS AGENTS

- Autonomous Agent: as any software that assists people and acts on their behalf.

- Example: in Network Management, Autonomous Agents could help in automating repetitive tasks, remembering rules that users frequently forget and summarizing complex data in understandable human-friendly reports or alerts.

3. INTELLIGENT AUTONOMOUS AGENTS

Framework for a Generic Agent

- INFERENCE module: which implements the Rule Deduction Engine and also stores the knowledge database. In future systems, it will hold the Artificial Neural Network part that is nowadays separated in customized Neural Agents.

3. INTELLIGENT AUTONOMOUS AGENTS

Framework for a Generic Agent

- INTERACTION module, which performs the interface between the agent and the environment.

3. INTELLIGENT AUTONOMOUS AGENTS

Framework for a Generic Agent

- COMMUNICATION module , which implements the message exchange procedures between agents. Communications are implemented by means of Knowledge Query and Manipulation Language (KQML) [10], which was chosen due to its strong commitment to agent applications.

3. INTELLIGENT AUTONOMOUS AGENTS

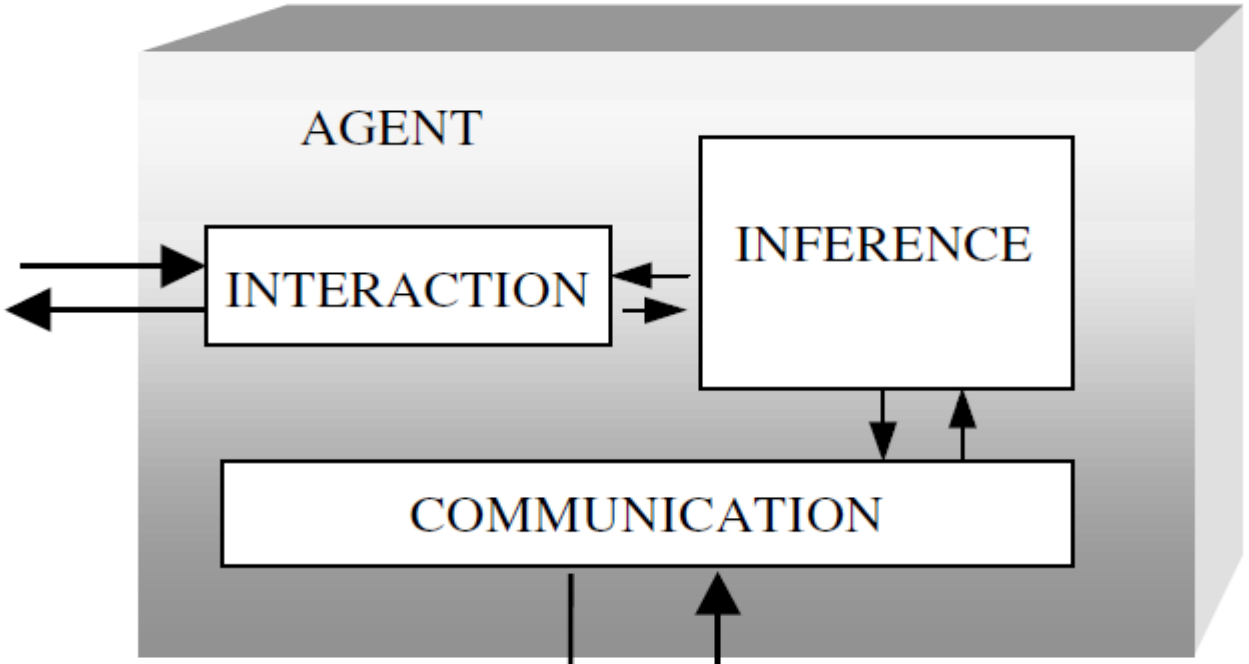
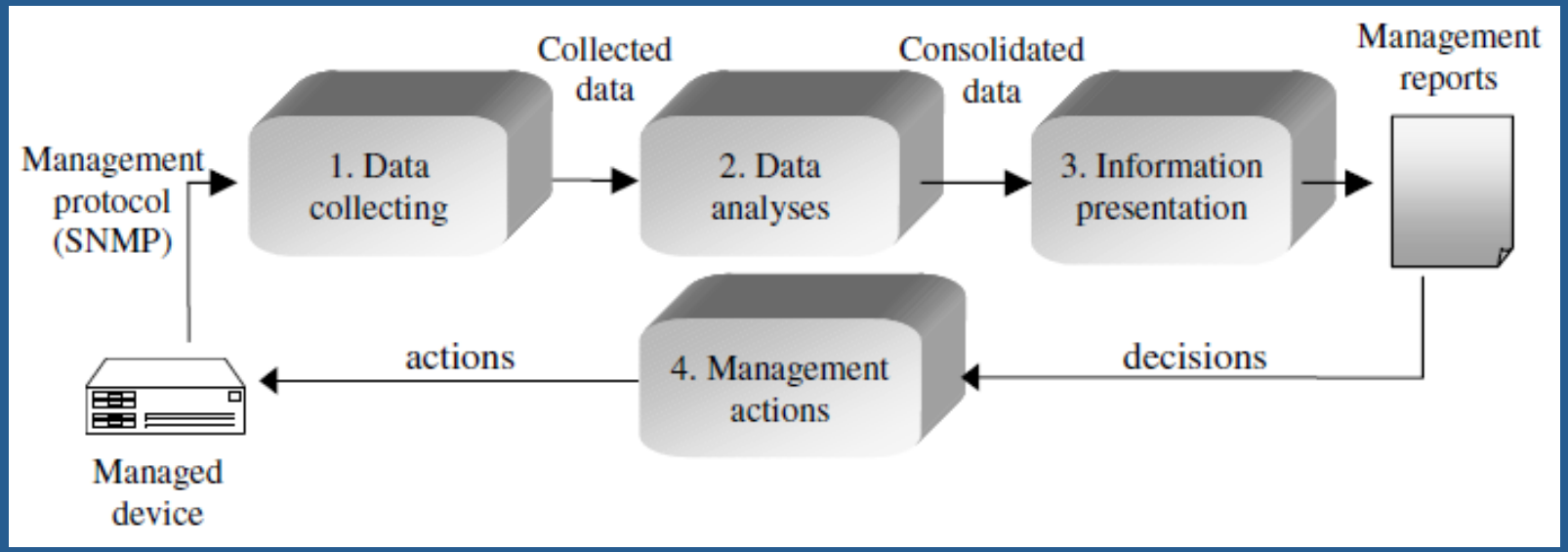


Fig. 1. Generic agent structure.

4. NETWORK MANAGEMENT AND AUTONOMOUS AGENTS

- A common network management provides a workflow like this:



- Data> ->Reports -> decisions -> actions

4. AUTONOMOUS AGENTS

- AgP: Collector Agent
- AgM: Manager Agent
- AgI: Interface Agent
- AgF: Facilitator Agent

4. DATA COLLECTION AGENTS (AgP)

- Retrieves data from devices and saves this information on local knowledge databases;

- "Knowledge" kept and processed locally, "shared" through the Agents Communication module;
- Most important part : Collecting Data is implemented by special knowledge rules loaded inside knowledge database;

4. MANAGEMENT AGENTS(AgM)

- Functions: Responsible for the organization of operation between agents of the same community. They can implement a higher level of data analyses based on shared knowledge (data) coming from different Collector Agents and coordinate multiple Data Collector Agents.

4. MANAGEMENT AGENTS(AgM)

Agents for Network Management:

- *Data Analyzers:* analyze data derived from different knowledge bases;
- *Data Consolidators:* combine or correlate data coming from distinct Data Collector Agents,
- *Future Value Predictors:* Neural Network structures for Time Series Prediction;
- *Value Classifiers:* data classification in scales, instead of linear values.

4. INTERFACE AGENTS(AgL)

- Bridge between the Agent Communities and human managers;
- E-mail, terminal-like and Web Interfaces;
- The use of appropriate Interface type depends on the kind of data output the manager needs;

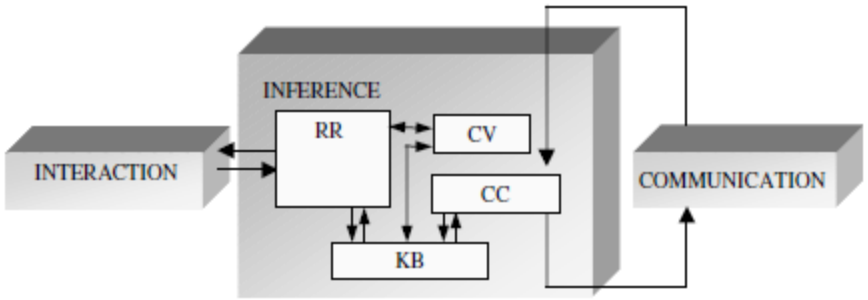
4. FACILITATOR AGENTS(AgF)

- Are agents needed in an Agent community to work as brokers for the information exchange process;
- Useful for data exchanging;
- Examples : List of available service names, routing, mediation and translation;

5. IMPLEMENTATION

- -Implementation
- -Detail for inference module
- -Data Collector Internal Knowledge Base

Koch and Westphall



RR: Rule Resolution Component
CV: Life cycle Component
CC: Cooperation Component
KB: Knowledge Base

Fig. 4. Detail for INFERENCE module.

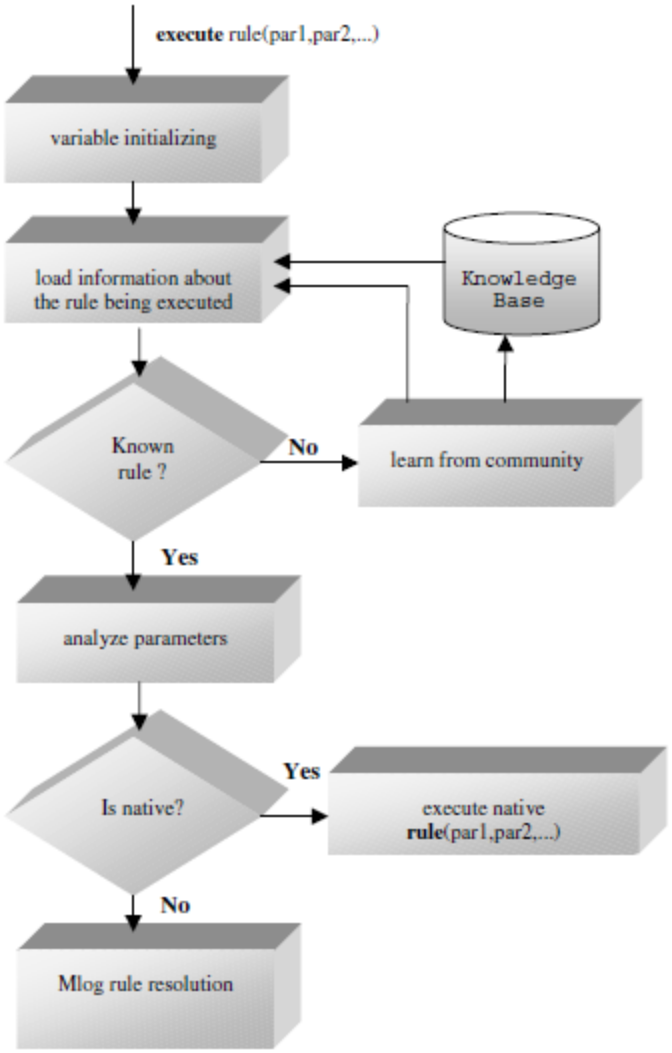


Fig. 5. Basic rule resolution flowchart.

```
INTERFACE module
SNMPget(IPAddress, OID, Value) :- native.
SNMPset(IPAddress,OID,Value) :- native.

COGNITIVE module
// *** Knowledge base (KB)
MIB("ifInOctets"," 1.3.6.1.2.1.2.2.1.10");
MIB("ifOutOctets"," 1.3.6.1.2.1.2.2.1.16");
Equipment("router","194.0.0.10").
Equipment("hub","194.0.0.11").
EquipmentMIBVar("router","ifInOctets",".0").
EquipmentMIBVar("hub"," ","ifInOctets",".0").
// *** Production rules (PR)
SetVars() :- setGlobal(DBName, $AgName+$Time+".db").
CollectDataEquipment(IPAddress, OID, Value) :-
       SNMPget(IPAddress, OID, ?Value).
SaveData(Time, EquipName, DescrMIB, Value) :-
       Assert("data",$Time, EquipName, DescrMIB, Value);
MonitorNetwork():-
       ?Equipment(?EquipName, ?IPAddress),
       ?EquipmentMIBvar(EquipName, ?MIBName, ?Interface),
       ?MIB(MIBName,?OID),
       ?CollectDataEquipment(IPAddress, OID+Interface, ?Value),
       StoreDataDB($DBName, Time, EquipName, MIBName, Value).

// *** Life cycle rules (CV)
On Start run SetVars().
Each 1min run MonitorNetwork().
Each 1hour run SaveDB($DBName).

// *** Cooperation Component (CC)
On UnknownRule($Rule) event run if(learn($Rule,30secs),resume,cut).
On InvalidValue event run cut.
On InternalError event if(gc(),resume,die).
```

* Of course, this is external representation and doesn't reflect internal knowledge storage structures inside Agents. We opted for a Prolog like output for sake of comprehensibility.

6. CONCLUSION

Autonomous Agents for Network Management are still being discovered, and the possibilities to explore actually Agent technology are countless.