DARPA

ACIP

COGnitive ENGine Technologies (COGENT) – An Innovative Architecture for Cognitive Processing

Julius Bogdanowicz	Raytheon - Principal Investigator
John Granacki	USC-ISI - Co-Principal Investigator

September 20, 2006

This work was sponsored by DARPA IPTO in the Architectures for Cognitive Information Processing program with Dr. William Harrod as Program Manager; contract number FA8650-04-C-7126

Raytheon ISI Information Sciences Institute Mercury Computer Systems, Inc. Exogi University of Pittsburgh HRL LABORATORIES CREST at GEORGIA TECH

Agenda

- COGENT Team
- Goals
- Study Approach
- Requirement Drivers
- HW Philosophy
- Architecture Levels
- Agent Based Cognitive System Model
- COGENT Hardware & Software
- Performance
- Differences with Conventional Architectures
- Summary

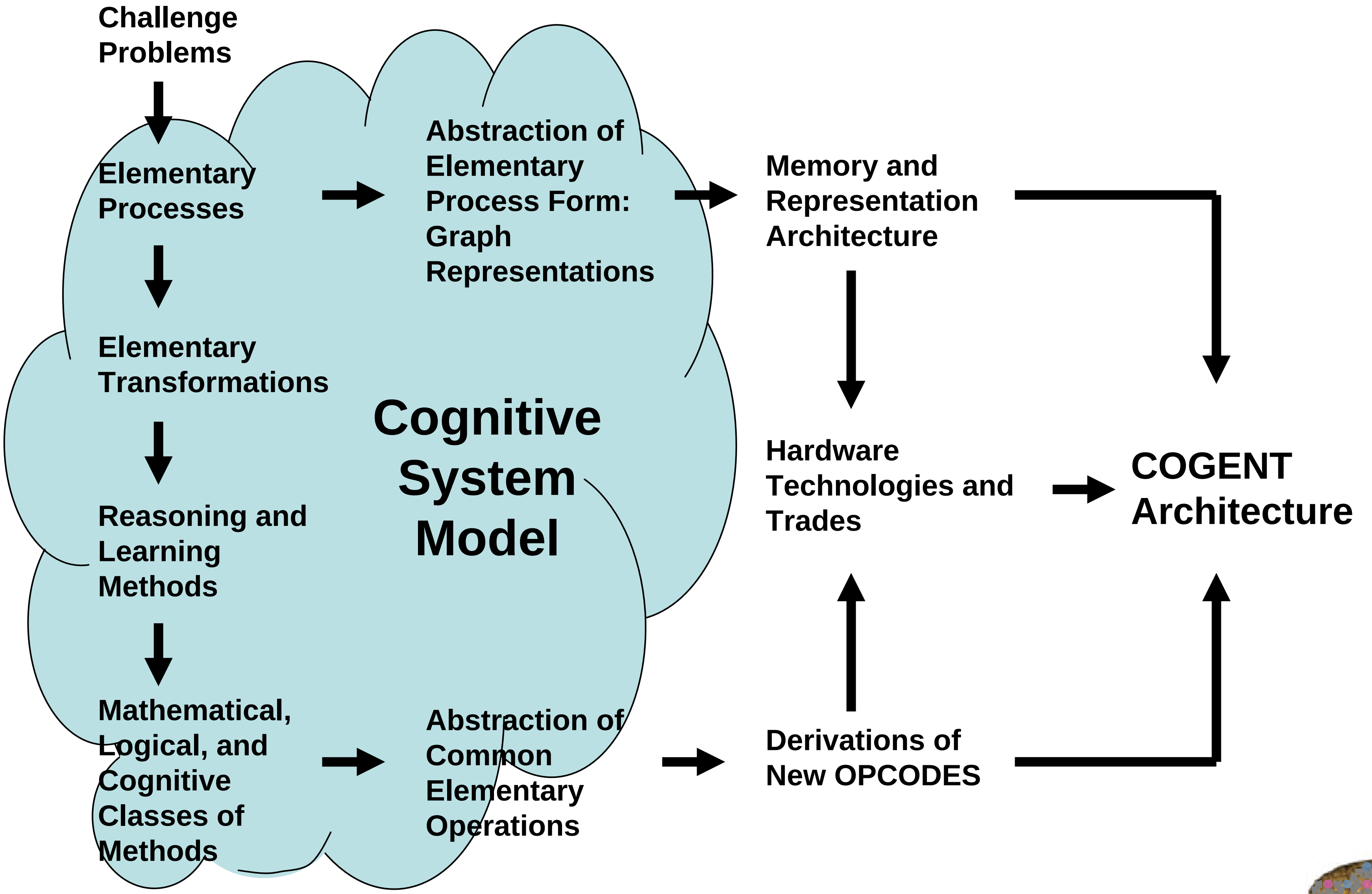
COGENT Team Members

- Raytheon: Julius Bogdanowicz, Michael Vahey, Brad Miller, Bradley Norman, Matt Benjamin, Mark Redekopp, Doug Brink
- USC-ISI: John Granacki, Jeff LaCoss, Wei-Min Shen, Andrew Gordon, Jerry Hobbs, Mark Moll, Behnam Salemi
- Exogi: Craig Steele
- Mercury Computer Systems: Jim Kulp
- University of Pittsburgh: Daniel Mosse, Bruce Childers, Jonathan Misurda
- HRL: Howard Neely, Michael Daily
- Georgia Tech: Sudhakar Yalamanchili, Krishna Palem, Vincent Mooney, Santosh Pande

COGENT Goals/Objectives

- Broad Cognitive System Model **based on spanning set of cognitive components that will efficiently implement current functions and enable new classes of cognitive algorithms**
- Scalable computational fabric with morphable, heterogeneous hardware engines **supporting multiple cognitive functions**
- Extensible, open architecture **allows general and special purpose accelerators for signal, data, and cognitive processing**
- Communications network **enables tight coupling of cognitive processing with classical signal, image, and data processing**
- Instrumented hardware architecture **for reacting to external environment and dynamic resource demands**
- Self awareness**, reacts to measured processor & memory activity patterns and the external environment to evaluate progress towards goals and achieves best results within time constraints**

Flow from Challenge Problems to Architecture



Challenge Problems

Elementary Processes

Elementary Transformations

Reasoning and Learning Methods

Mathematical, Logical, and Cognitive Classes of Methods

Abstraction of Elementary Process Form: Graph Representations

Cognitive System Model

Abstraction of Common Elementary Operations

Memory and Representation Architecture

Hardware Technologies and Trades

Derivations of New OPCODES

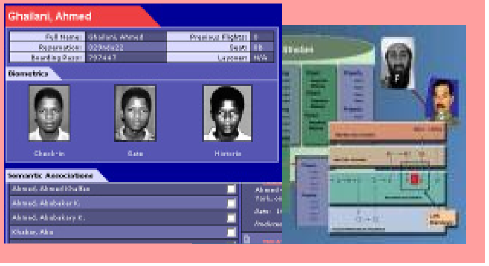
COGENT Architecture

Approved for Public Release, Distribution Unlimited

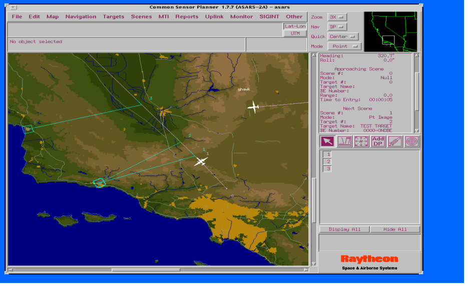
Cognitive Problems Identify Computing Needs

Human Computer Interface

Intelligence Analysis



UAV Dynamic Planner



- **Process cognitive applications for military missions**
  - *Recognition* of warfighter intent
    - Understanding of warfighter desires in context
    - Interaction driven by cognitive agents intentions
    - ***Human problem understanding & decision making markedly improved***
  - *Analysis* of intelligence data
    - Detection of hidden relationships in very large knowledge bases
    - Slowly changing knowledge base
    - ***Process very large problems***
  - *Planning* for wide range of missions
    - Single autonomous vehicles → battlefield
    - Rapidly changing working data
    - ***Deliver real-time response in highly dynamic environments***
- **Lessons learned**
  - Applications need a robust Cognitive System Model
    - Adopted an Observe - Orient - Decide – Act + Learn (OODA+L) model based on a combination of research cognitive models
    - Need latency tolerant processing techniques with large memories
    - Need sophisticated memory management techniques for episodic and long term memory
  - Confirmed hardware support needed for agents, graphs, Bayesian networks & a wide range of computational kernels
  - To simplify application development we decouple the computational view from developers view
  - *New classes of algorithms are required to exploit the new computational fabric; must re-think the underlying computational model*

Cognitive Motivation

- ***Cognitive applications are characterized by*:**
  - Graph based operations and data structures
  - Sparse knowledge representation
  - “Inexact” Information
  - Very large amounts of parallelism at multiple levels
    - **Observe**: Input symbols distributed to multiple agents (***sub/pub***)
    - **Orient/Decide**: Competing Possible Worlds (***OR-parallelism***)
    - **Orient/Decide**: Searching and matching (***Graph parallelism***)
  - Potential for speculative processing – multiple predictive processes
  - Approximate solutions provided by “anytime” and best-available calculations
    - ***Prioritize*** promising processing contexts
    - ***Filter/Prune*** stale (too late) and ineffective (poor solution) processing
  - Learning - dynamic additions to knowledge base
- ***Cognition is poor match to conventional systems***
  - ***Limited parallelism with user specified management***
  - **Memory-intensive**
    - Extensive pointer-chasing through graphs
    - Memory access is data dependent, limiting effective use of data caches
      - Profiling experiment: observed 1 IPC on 4-issue SGI system (80% data cache miss)
  - Processors optimized for numeric, not symbolic processing

COGENT HW Architecture Philosophy

- **Enable & exploit parallelism at all levels**
  - Multiprocessor system with very large distributed memory
  - HW generates and manages parallelism
    - Independent agents and/or Possible Worlds running in parallel
    - HW-managed multicasting of inputs to agents via sub/pub mechanism
    - Graph operations spawn parallel search and match operations
  - Mitigate cost of speculative computations
  - *Minimize user awareness of parallelism*
- **Manage parallel processing in HW where possible**
  - Prioritizing – “promote” promising threads
  - Filtering – quickly prune ineffective and “too late” threads
  - Synchronizing – enforce “check-in”
  - *Enable anytime or earliest/best processing*
- **Provide HW support for cognitive middleware**
  - Graph/Bayesian/HMM data structures
  - Fast access of distributed objects, network routing, etc.

COGENT Architecture Levels

- Specify Mission goals & requirements
- Performs goal selection, QoS

- HL OODA+L Cognitive Architecture with problem-solving rules specific to app

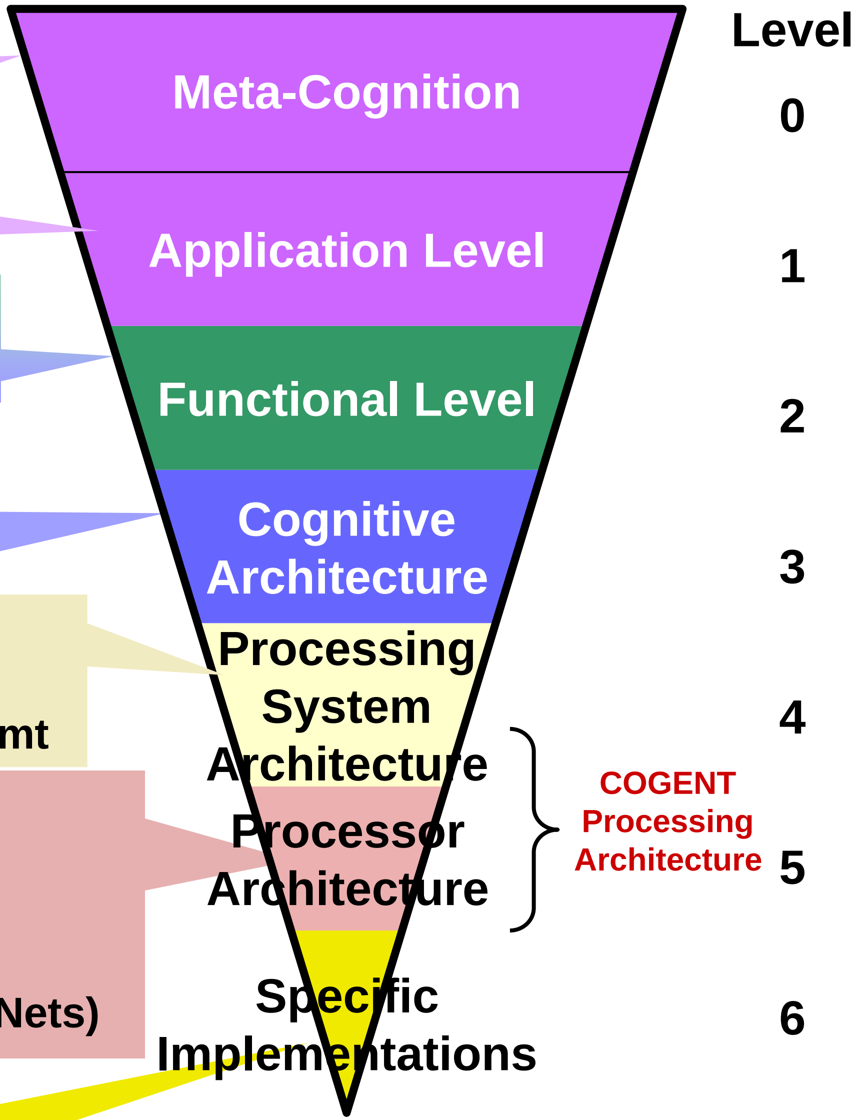
- Cognitive Algorithms & Agent Architecture
  - Newell 7+2

- Common Cognitive Services
  - Cognitive Agent Oriented Programming
  - Kernels (e.g., Probabilistic Reasoning)

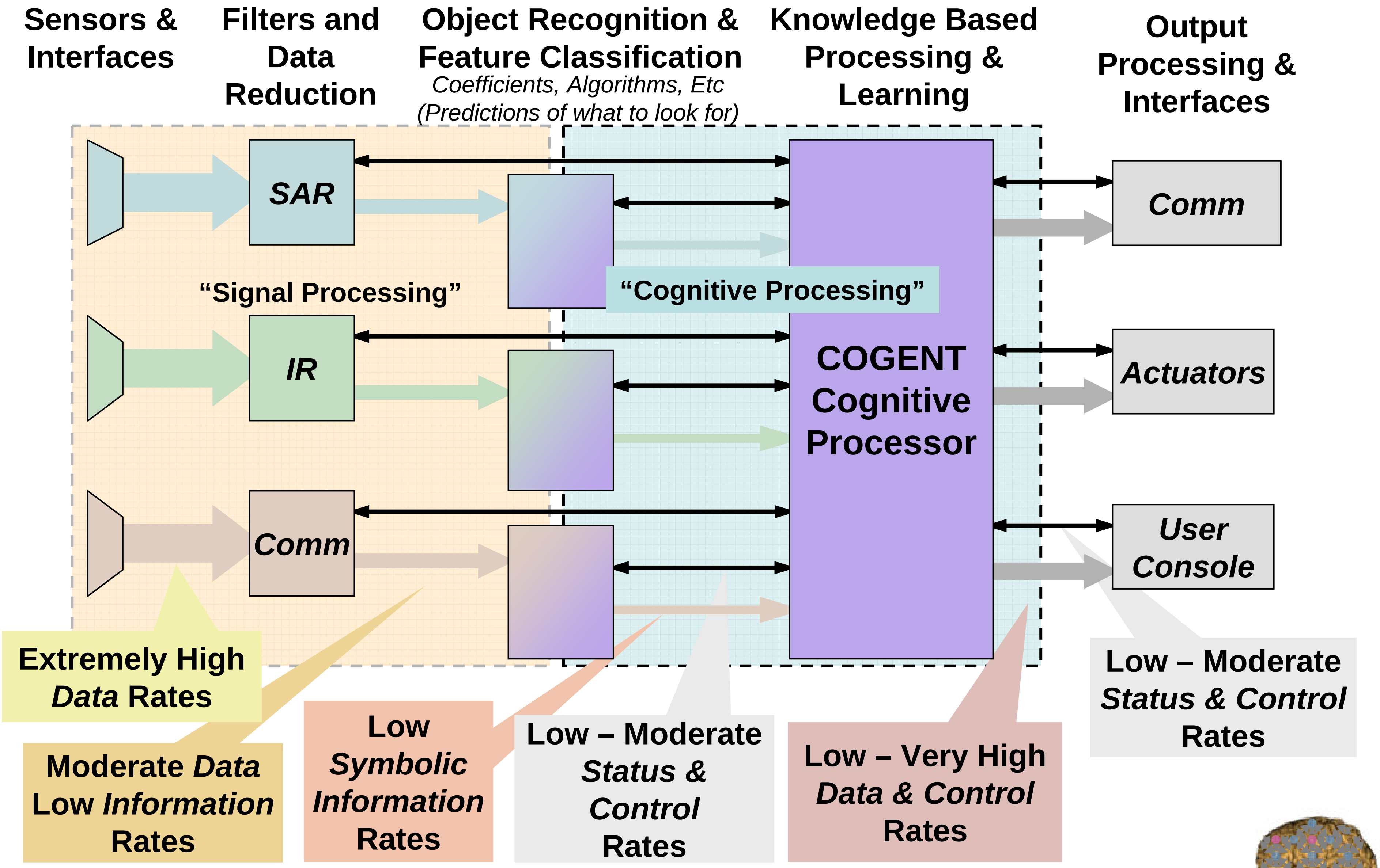
- Run-time High Level Compiler
- Abstract Machine Models
- Run-time Low Level Compiler & Resource Mgmt

- Machine organization
  - Single Name Space
- Hardware micro-architectures
  - Graphs
  - Special-Purpose Accelerators (e.g. Bayes Nets)

- Optimized cognitive systems
  - Servers
  - Embedded/real-time

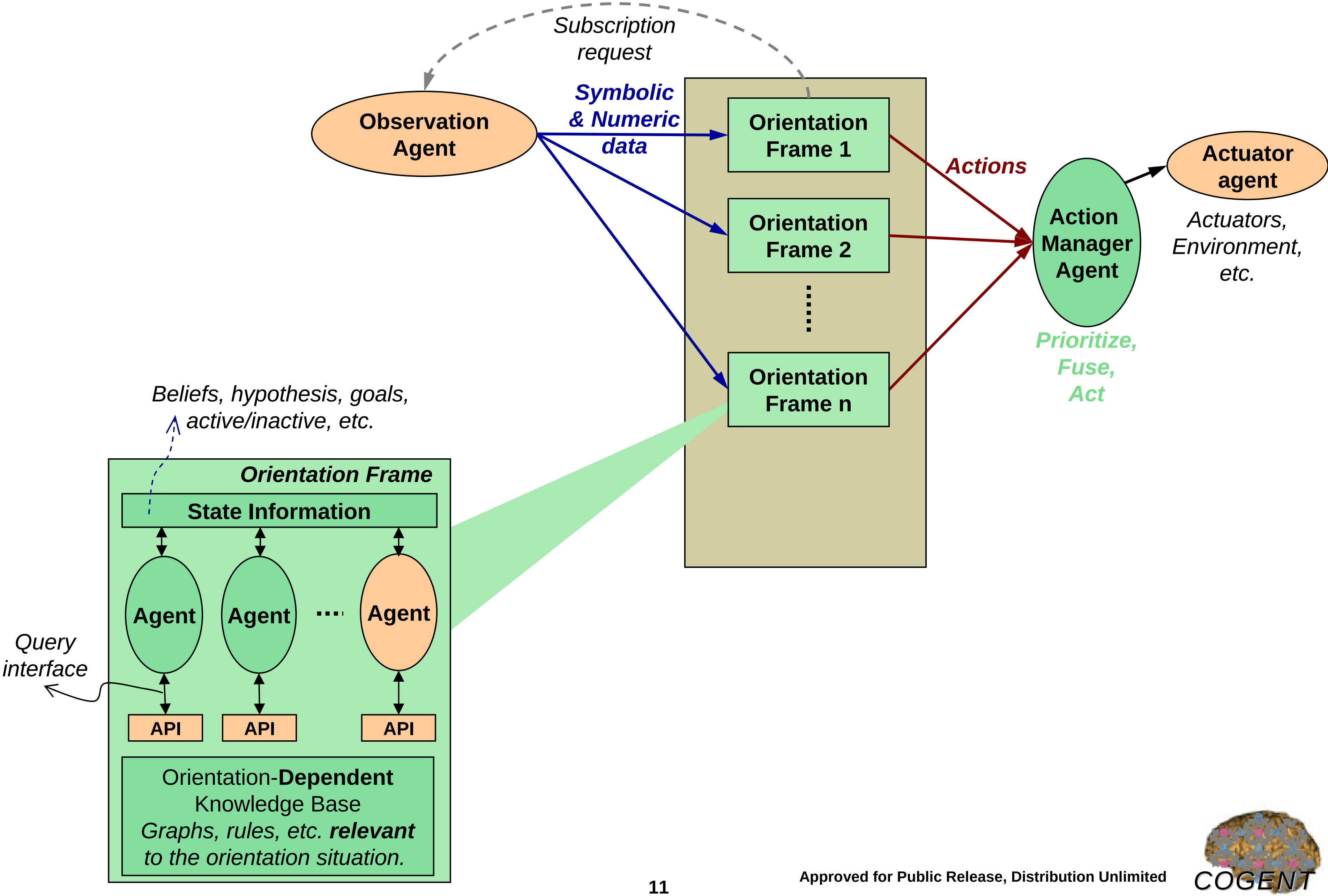


COGENT Hardware and Functional Organization for Embedded Applications


Sensors & Interfaces
Filters and Data Reduction
Object Recognition & Feature Classification
Coefficients, Algorithms, Etc
(Predictions of what to look for)
Knowledge Based Processing & Learning
Output Processing & Interfaces
SAR
IR
Comm
"Signal Processing"
"Cognitive Processing"
COGENT Cognitive Processor
Comm
Actuators
User Console
Extremely High Data Rates
Moderate Data Low Information Rates
Low Symbolic Information Rates
Low – Moderate Status & Control Rates
Low – Very High Data & Control Rates
Low – Moderate Status & Control Rates

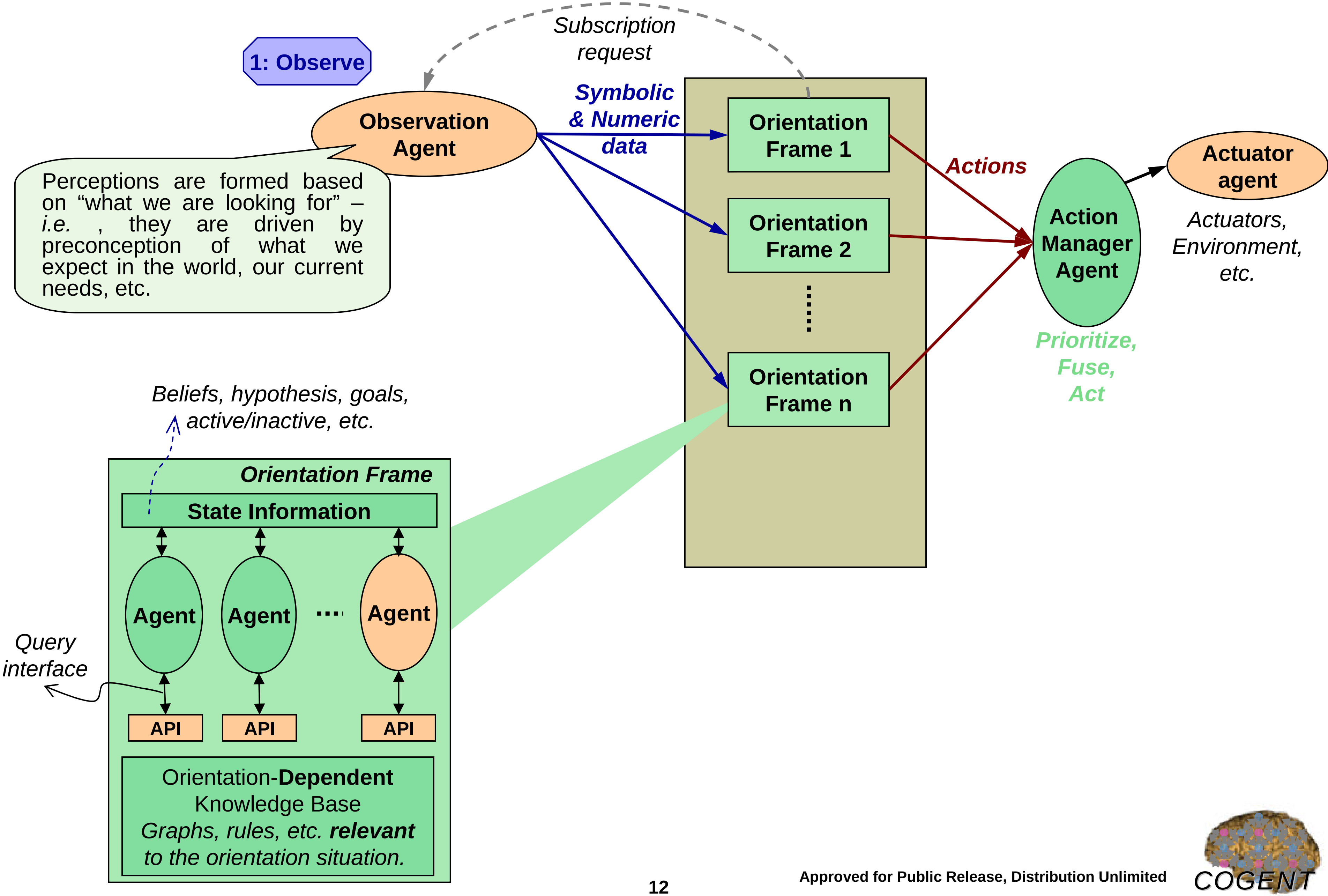
Approved for Public Release, Distribution Unlimited

We Implement the Cognitive Architecture Using Intelligent Agents



Approved for Public Release, Distribution Unlimited

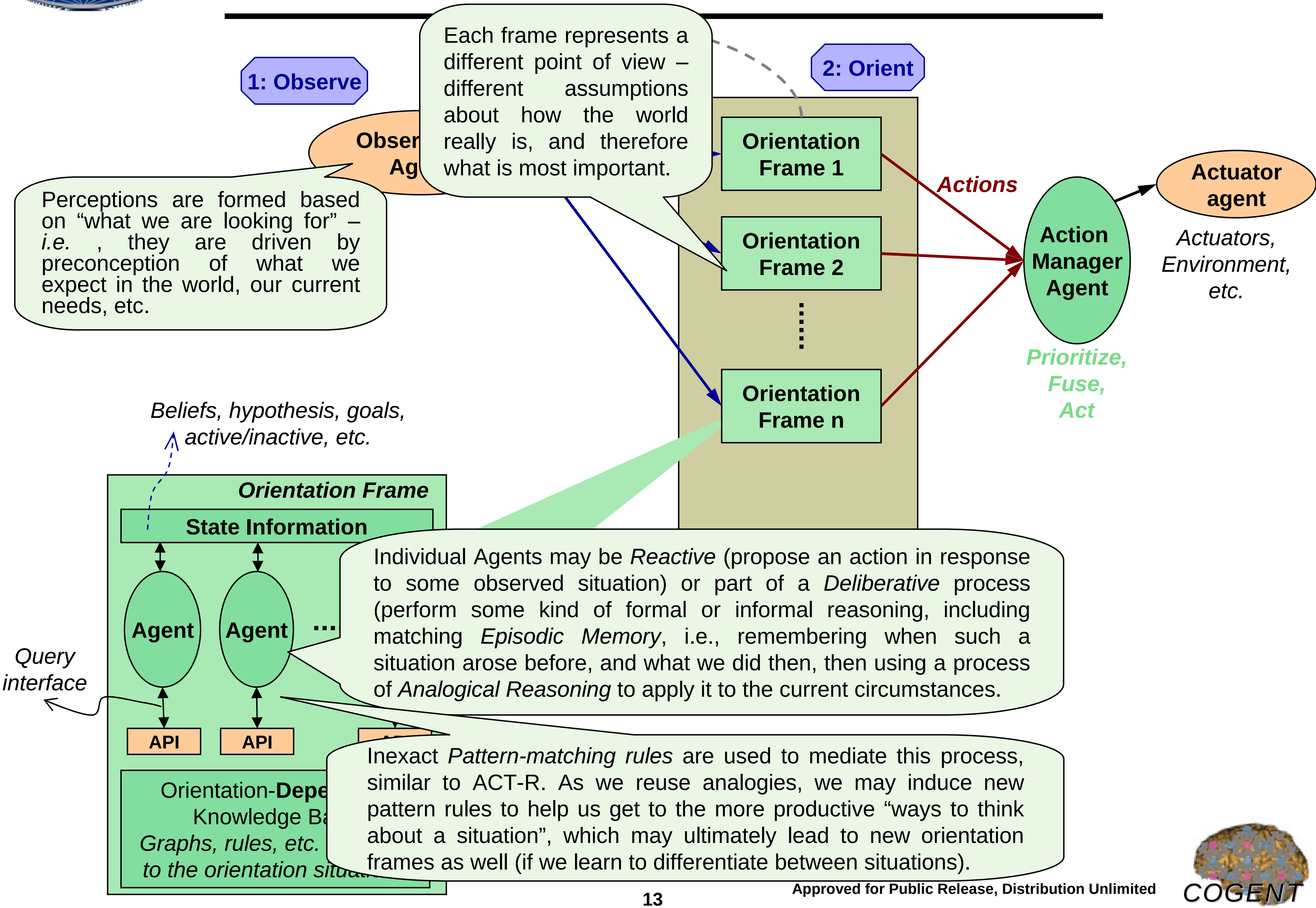
We Implement the Cognitive Architecture Using Intelligent Agents



Perceptions are formed based on “what we are looking for” – *i.e.* , they are driven by preconception of what we expect in the world, our current needs, etc.

Approved for Public Release, Distribution Unlimited

We Implement the Cognitive Architecture Using Intelligent Agents



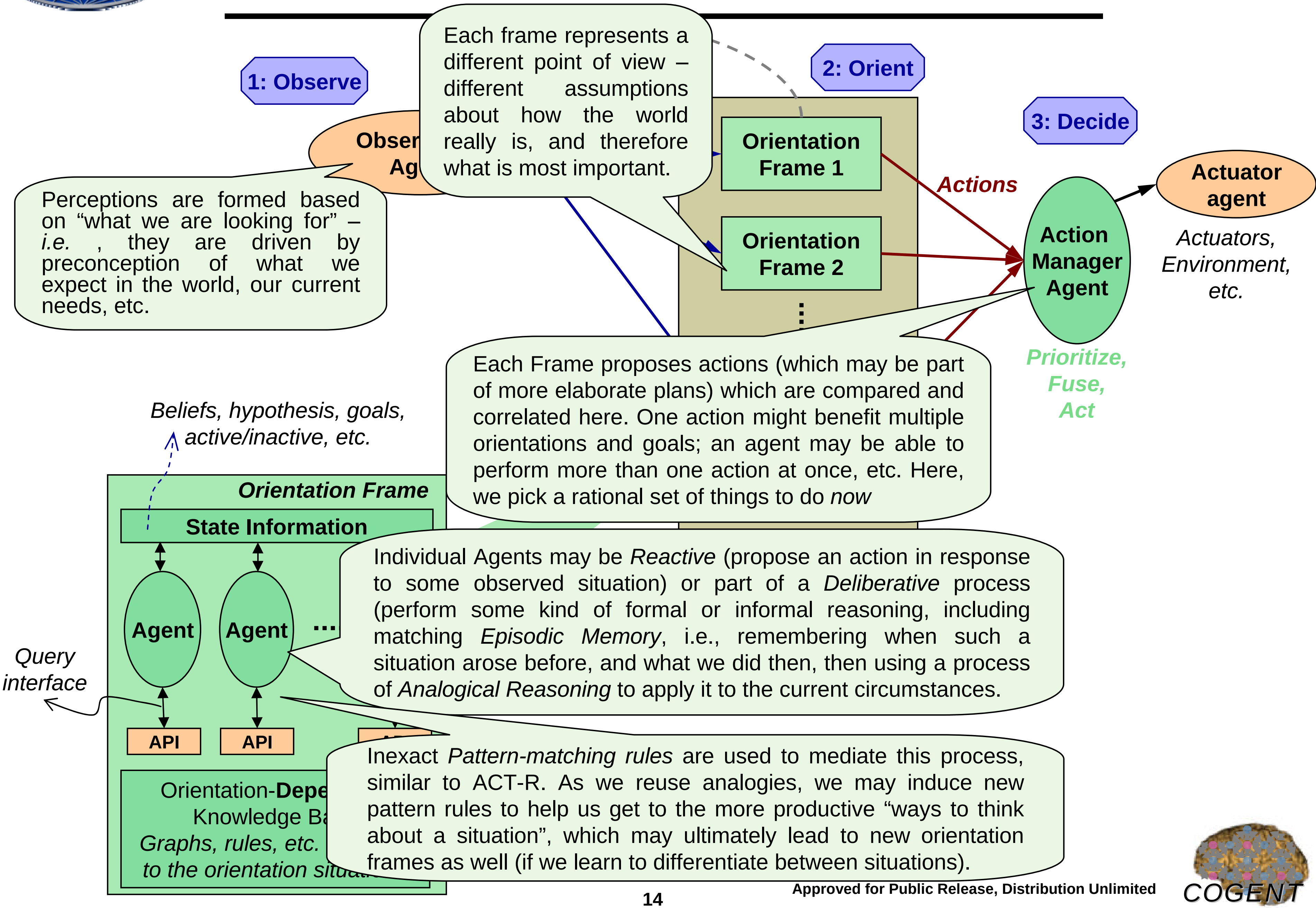
Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

Perceptions are formed based on “what we are looking for” – *i.e.* , they are driven by preconception of what we expect in the world, our current needs, etc.

Individual Agents may be *Reactive* (propose an action in response to some observed situation) or part of a *Deliberative* process (perform some kind of formal or informal reasoning, including matching *Episodic Memory*, i.e., remembering when such a situation arose before, and what we did then, then using a process of *Analogical Reasoning* to apply it to the current circumstances.

Inexact *Pattern-matching rules* are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive “ways to think about a situation”, which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).

We Implement the Cognitive Architecture Using Intelligent Agents



Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

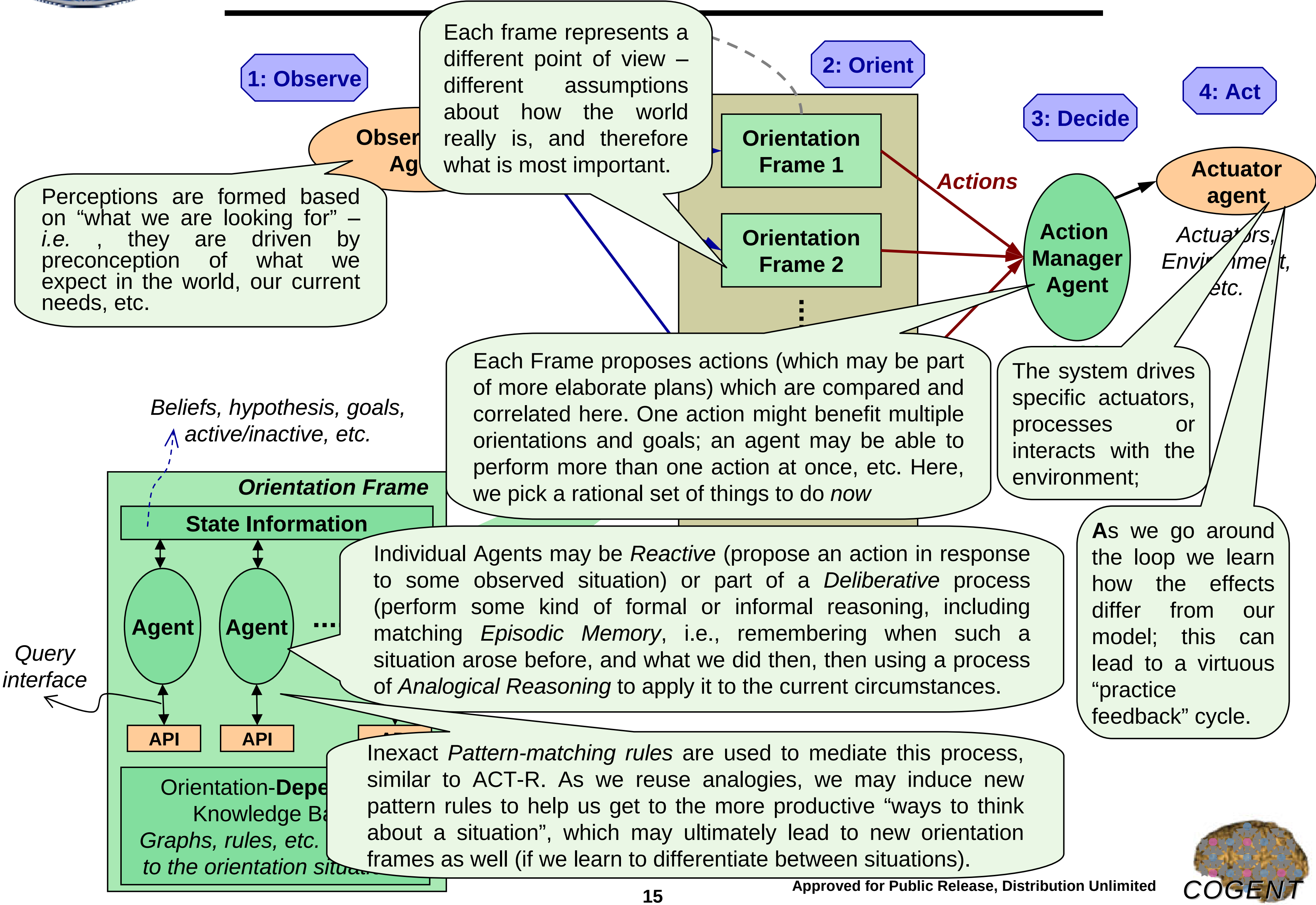
Perceptions are formed based on "what we are looking for" – *i.e.* , they are driven by preconception of what we expect in the world, our current needs, etc.

Each Frame proposes actions (which may be part of more elaborate plans) which are compared and correlated here. One action might benefit multiple orientations and goals; an agent may be able to perform more than one action at once, etc. Here, we pick a rational set of things to do *now*

Individual Agents may be *Reactive* (propose an action in response to some observed situation) or part of a *Deliberative* process (perform some kind of formal or informal reasoning, including matching *Episodic Memory*, i.e., remembering when such a situation arose before, and what we did then, then using a process of *Analogical Reasoning* to apply it to the current circumstances.

Inexact *Pattern-matching rules* are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive "ways to think about a situation", which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).

We Implement the Cognitive Architecture Using Intelligent Agents



Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

Perceptions are formed based on “what we are looking for” – *i.e.* , they are driven by preconception of what we expect in the world, our current needs, etc.

Each Frame proposes actions (which may be part of more elaborate plans) which are compared and correlated here. One action might benefit multiple orientations and goals; an agent may be able to perform more than one action at once, etc. Here, we pick a rational set of things to do *now*

The system drives specific actuators, processes or interacts with the environment;

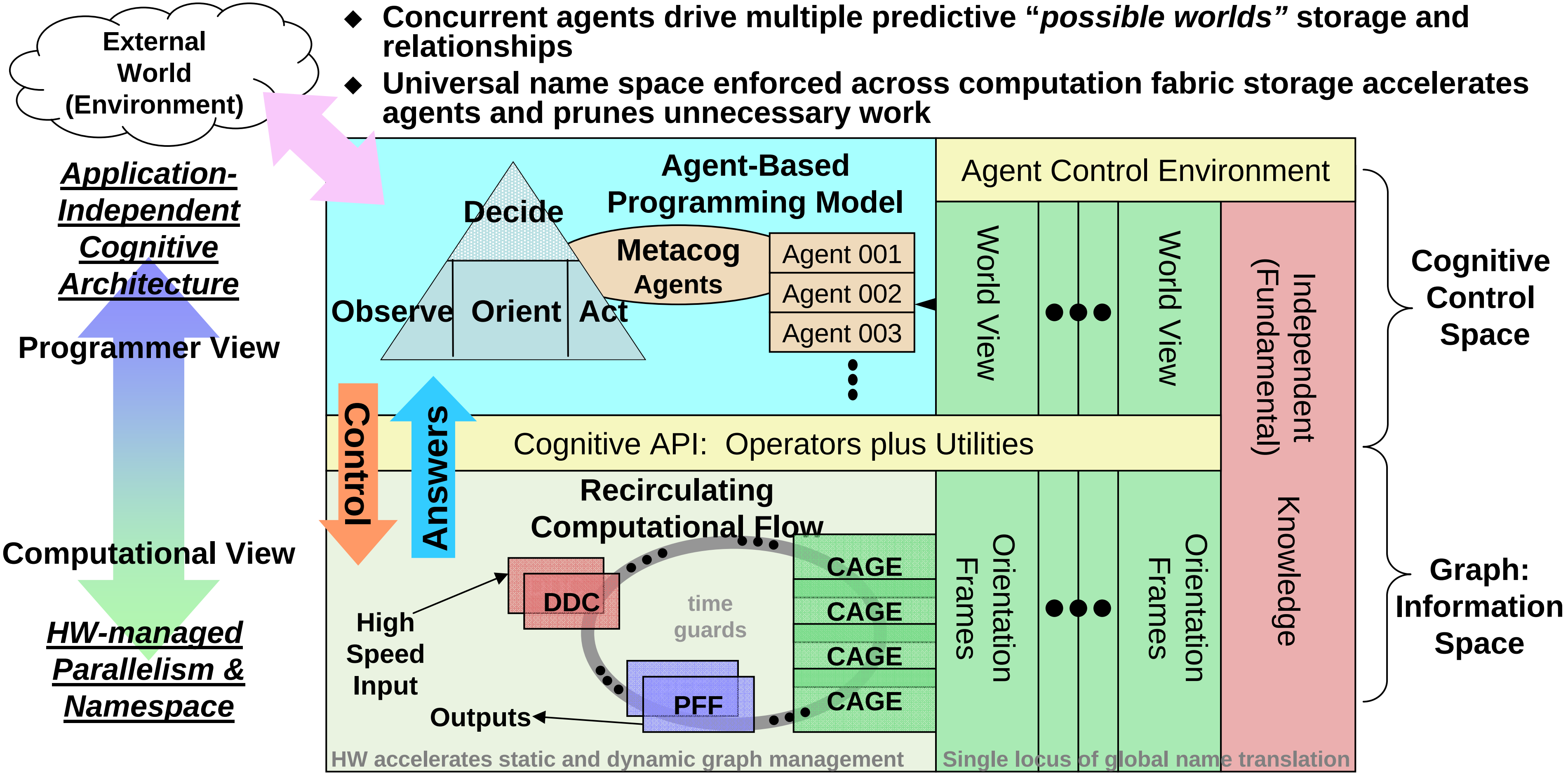
Individual Agents may be *Reactive* (propose an action in response to some observed situation) or part of a *Deliberative* process (perform some kind of formal or informal reasoning, including matching *Episodic Memory*, i.e., remembering when such a situation arose before, and what we did then, then using a process of *Analogical Reasoning* to apply it to the current circumstances.

As we go around the loop we learn how the effects differ from our model; this can lead to a virtuous “practice feedback” cycle.

Inexact *Pattern-matching rules* are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive “ways to think about a situation”, which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).

COGENT: A Highly Parallel Cognitive Processor

- Concurrent agents drive multiple predictive “*possible worlds”* storage and relationships
- Universal name space enforced across computation fabric storage accelerates agents and prunes unnecessary work


External World (Environment)
Application-Independent Cognitive Architecture
Programmer View
Computational View
HW-managed Parallelism & Namespace
Agent-Based Programming Model
Decide
Observe
Orient
Act
Metacog Agents
Agent 001
Agent 002
Agent 003
Agent Control Environment
World View
World View
Independent (Fundamental) Knowledge
Cognitive Control Space
Control
Answers
Cognitive API: Operators plus Utilities
Recirculating Computational Flow
DDC
time guards
CAGE
CAGE
CAGE
CAGE
PFF
High Speed Input
Outputs
Orientation Frames
Orientation Frames
Graph: Information Space
HW accelerates static and dynamic graph management
Single locus of global name translation

DDC = Data Distribution Center- hardware for automatic “scatter” (distribution of computation to the relevant, distributed data)

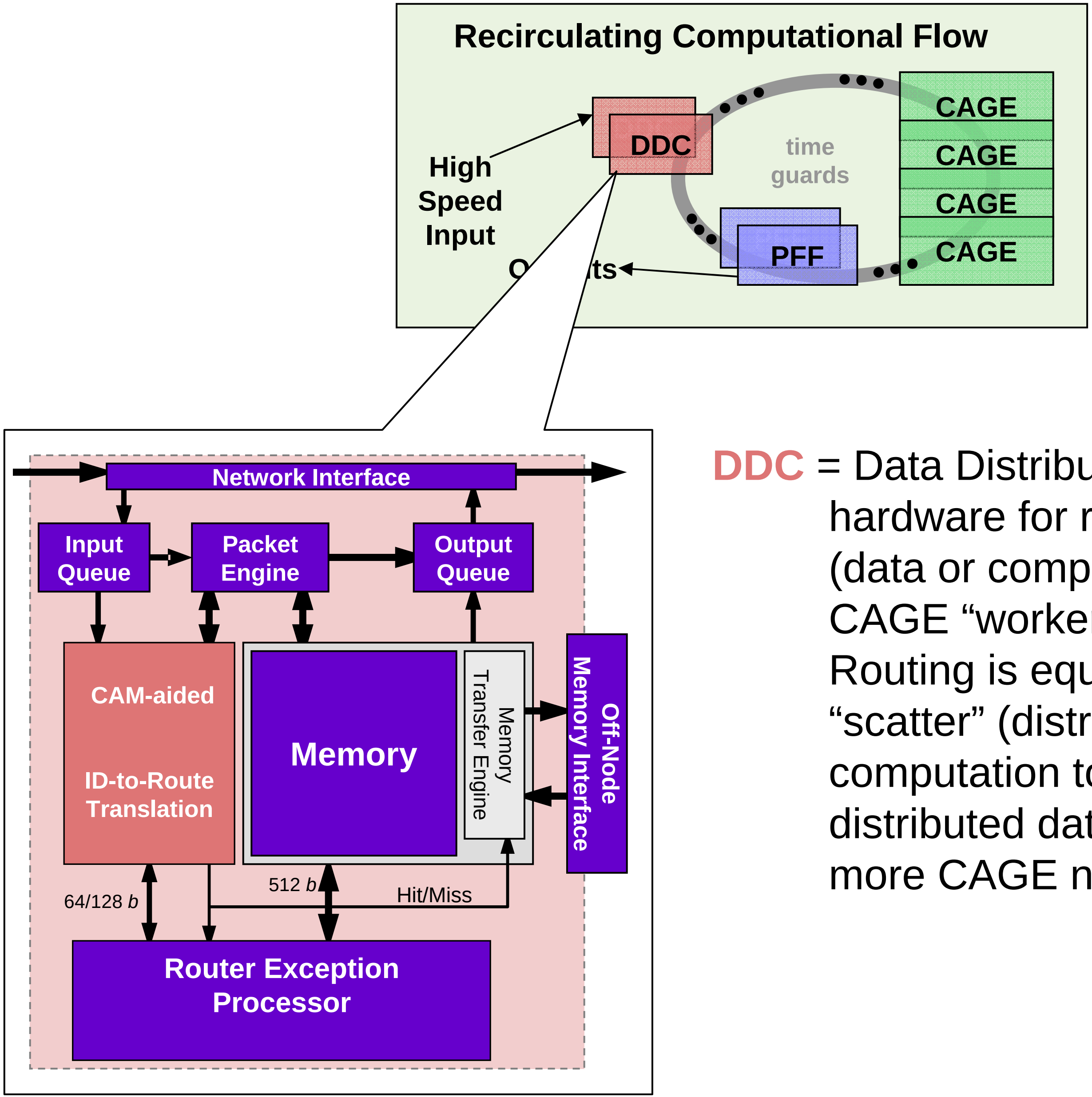
CAGE = Cognitive Agent & Graph Engine – form parallel computing fabric, accelerates primitive operations on graphs, supports probabilistic reasoning; uses distributed, scalable data storage

PFF = Prioritize, Filter and Fuse – automatic gather, coalescing results, pruning of stale and unproductive computation, time based to assure timely results

Graphs and Efficiency

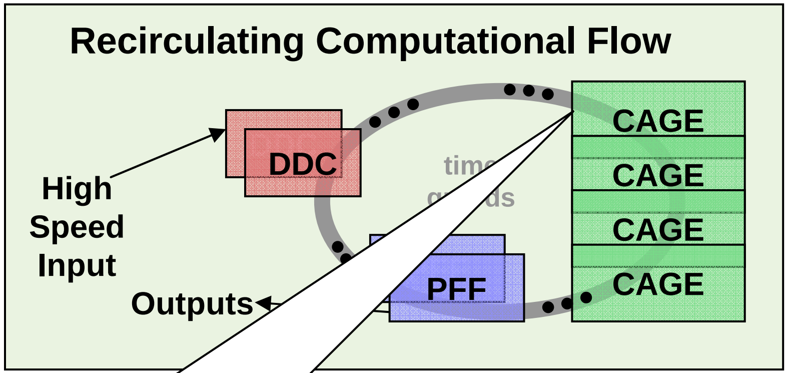
- **Highly parallel computation**
  - Relational structure of graphs makes inherent parallelism apparent
- **COGENT accelerates computation on large graphs:**
  - Program instructions sent to location of data to minimize data movement ➔ optimizes bandwidth usage.
  - HW managed label-based routing of graphs and vertices (DDC)
  - Fast lookup of local graph vertices on a CAGE node
  - Wide-word memory access & processing of graph components on a CAGE node
  - CAGE HW supports memory re-organization for efficient access to sparse to dense graph structures
    - Garbage collection performed during process

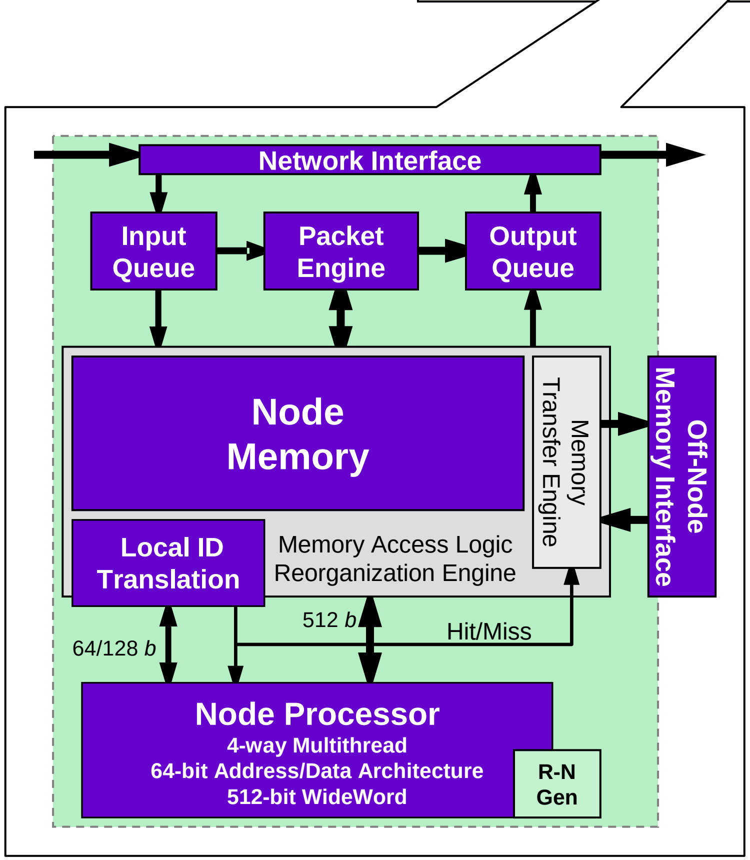
COGENT HW Architecture



DDC = Data Distribution Center - hardware for routing messages (data or computation tasks) to CAGE “worker” processors. Routing is equivalent to automatic “scatter” (distribution) of computation to the relevant, distributed data located in 1 or more CAGE nodes

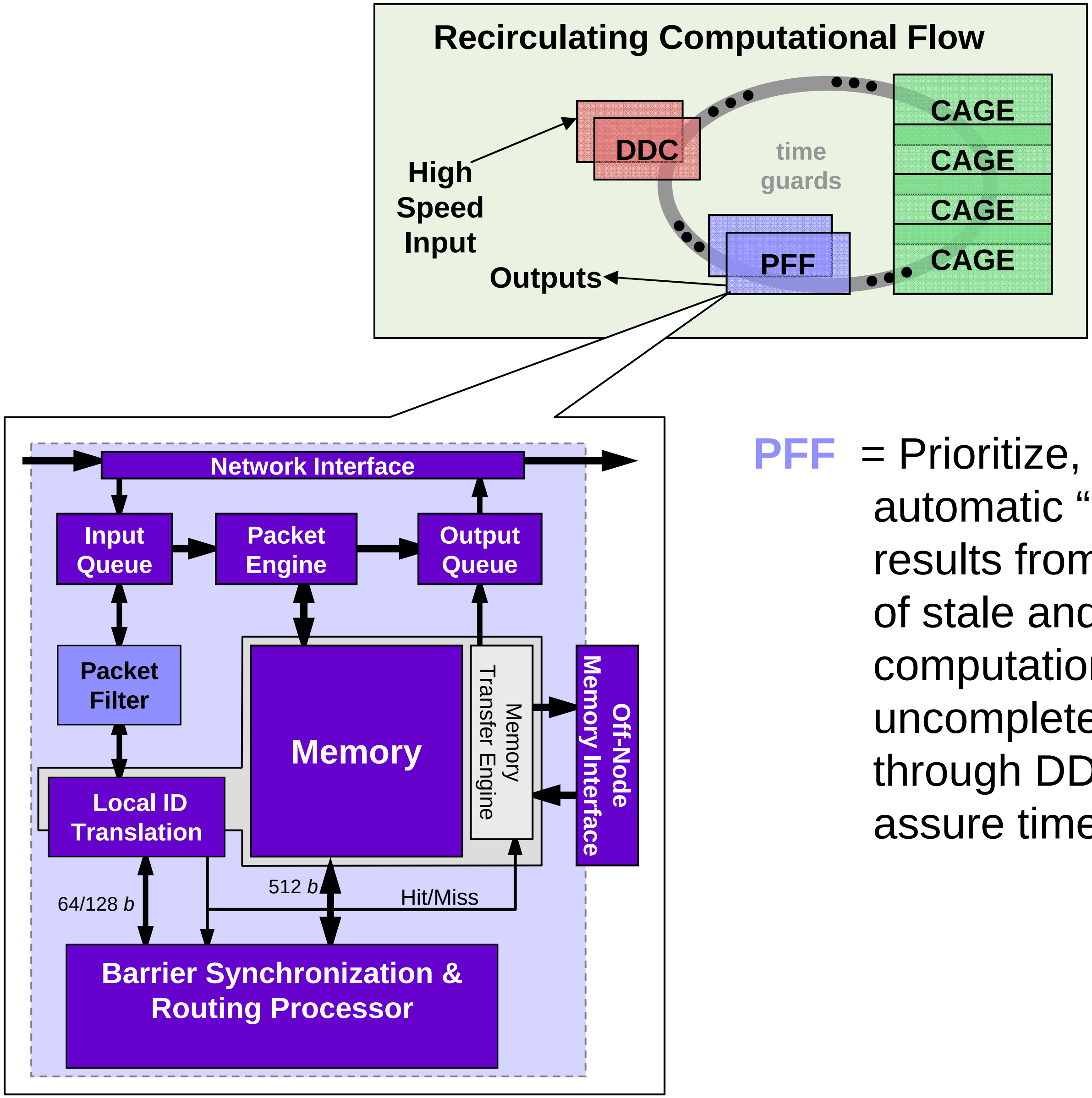
COGENT HW Architecture





CAGE = Cognitive Agent & Graph Engine(s) – form parallel computing fabric, hosts agents, augmented ISA accelerates primitive operations on graphs, supports probabilistic reasoning, uses distributed, scalable data storage; Agents distributed to 1 or more CAGE nodes, Agents spawn cognitive functions located on 1 or more CAGE nodes, Computations requests on non-local data are automatically sent to proper CAGE node via PFF & DDC

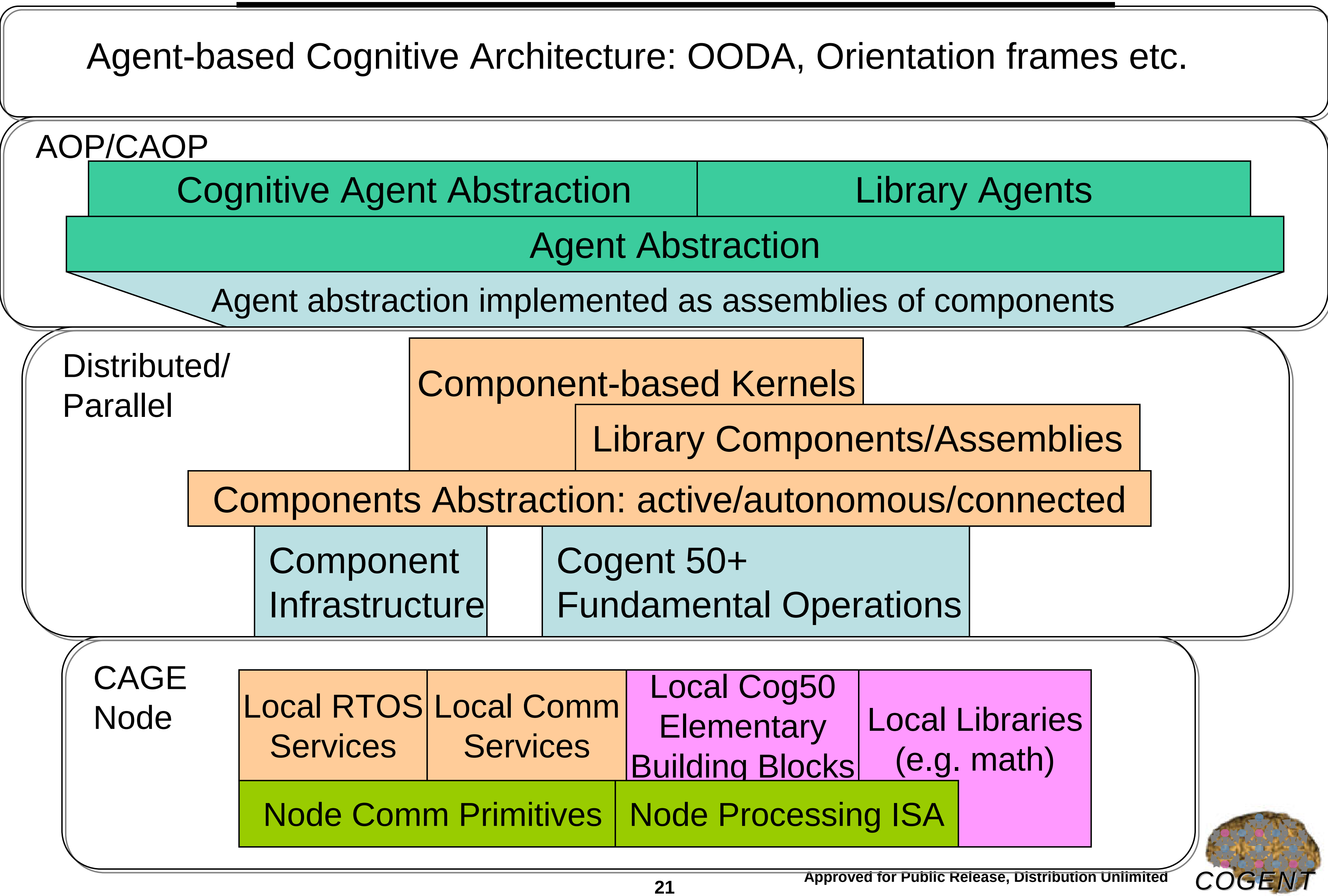
COGENT HW Architecture



PFF = Prioritize, Filter and Fuse – automatic "gather"/ coalescing results from CAGE(s), pruning of stale and unproductive computation, re-circulates uncompleted computations through DDC, time based to assure timely results

Approved for Public Release, Distribution Unlimited

Agents/Components



Agent-based Cognitive Architecture: OODA, Orientation frames etc.

AOP/CAOP

Cognitive Agent Abstraction

Library Agents

Agent Abstraction

Agent abstraction implemented as assemblies of components

Distributed/
Parallel

Component-based Kernels

Library Components/Assemblies

Components Abstraction: active/autonomous/connected

Component
Infrastructure

Cogent 50+
Fundamental Operations

CAGE
Node

Local RTOS
Services

Local Comm
Services

Local Cog50
Elementary
Building Blocks

Local Libraries
(e.g. math)

Node Comm Primitives

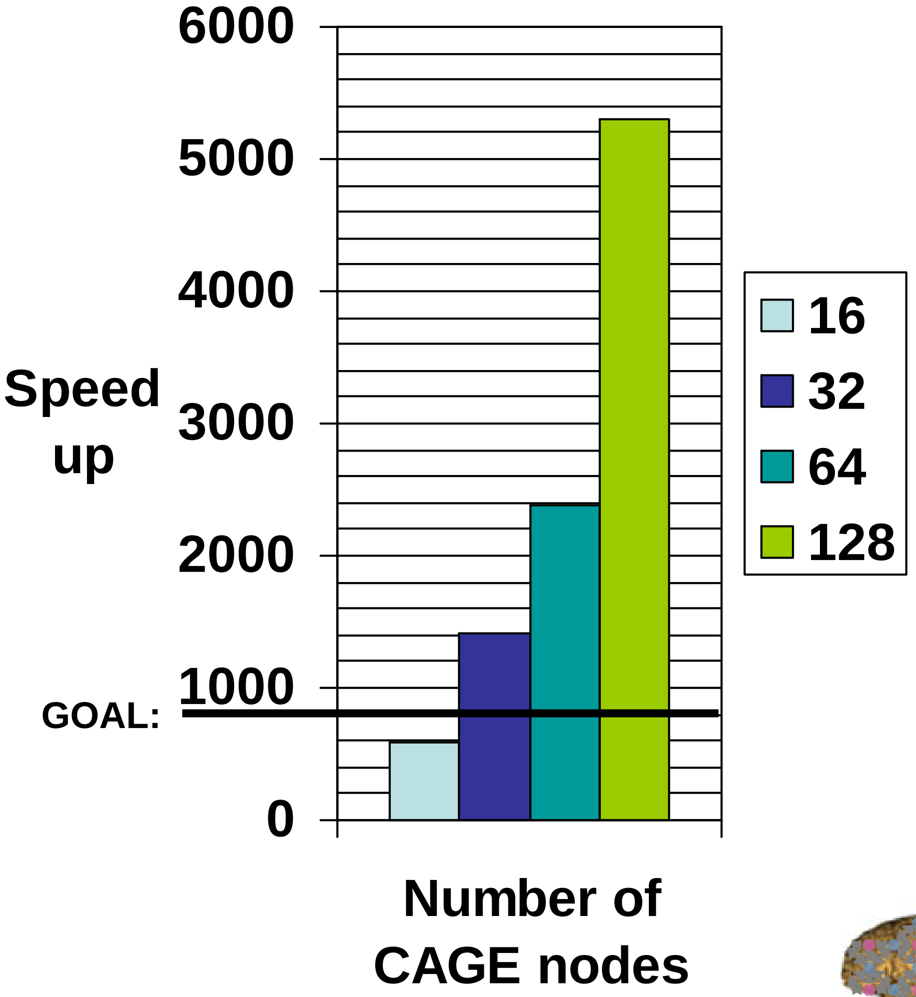
Node Processing ISA

Approved for Public Release, Distribution Unlimited

Example of Simulated Performance

- Surrogate for UAV ISR planner using analogical reasoning
- Problem size: 6600 propositions represented as a graph structure with 40K vertices & 80K edges
- Extrapolated performance on Intel Pentium 4 was 40 hours
- Goal: 800X speedup
- Minimum of 32 CAGE processors needed to exceed goal

Speed up Over Intel Pentium 4 Baseline



COGENT vs. Conventional Architecture

Conventional Processor	COGENT Processor
Relies heavily on cache: dynamic access patterns make cache ineffective	No cache – large knowledge bases won’t fit Large on chip memories with good BW – per processor memory BW > 100 GB/S
Load/store access to memory	HW manipulation of memory access, global ID vs. address, publish and subscribe to information sources
Word focused – no semantics	Semantically accessed memory: Graph based representation of knowledge – HW optimized access and manipulation mechanisms
General computing – RISC instruction set – register file focus – compiler driven program	Optimizations for the cognitive operations – intensive memory focus – staging and location of data HW optimized
Exact, repeatable deterministic functions – low level semaphores	Probabilistic representations, reconstructive memory – runtime synthesized data representations
Ops concept is driven from program counter, interrupts, etc.	Ops concept is driven from data flow and probabilistic data relationships – dynamically adjusted based on experience

Summary

- **COGENT is an innovative recirculating computational architecture for cognitive processing**
- **COGENT architecture is being driven by application & cognitive system model requirements**
- **A single unified hardware & software structure has been defined**
  - Hardware directly supports the management of parallelism
  - Agent based software structure provides foundation for OODA based cognitive system model to implement the applications
- **Simulations are being used to refine the architectural details**